2014

Human and Animal Behavior Understanding

Wenbin Chen
West Virginia University

Human and Animal Behavior Understanding

Wenbin Chen

Thesis submitted

to the Benjamin M. Statler College of Engineering and Mineral Resources

at West Virginia University

in partial fulfillment of the requirement for the degree of

Master of Science in Electrical Engineering

Guodong Guo, Ph.D., Chair

Donald Adjeroh, Ph.D.

Cun-Quan Zhang, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia

2014

Keywords: human/animal behavior recognition, RGB-D data, fusion

Abstract

Human and Animal Behavior Understanding

Wenbin Chen

Human and animal behavior understanding is an important yet challenging task in computer vision. It has a variety of real-world applications including human computer interaction (HCI), video surveillance, pharmacology, genetics, etc. We first present an evaluation of spatiotemporal interest point features (STIPs) for depth-based human action recognition, and then propose a framework call TriViews for 3D human action recognition with RGB-D data. Finally, we investigate a new approach for animal behavior recognition based on tracking, video content extraction and data fusion.

STIPs features are widely used with good performances for action recognition using the visible light videos. Recently, with the advance of depth imaging technology, a new modality has appeared for human action recognition. It is important to assess the performance and usefulness of the STIPs features for action analysis on the new modality of 3D depth map. Three detectors and six descriptors are combined to form various STIPs features in this thesis. Experiments are conducted on four challenging depth datasets.

After evaluating STIPs features for depth-based human action recognition, we propose an effective framework called TriViews to utilize 3D information for human action recognition. It projects the 3D depth maps into three views, i.e., front, side, and top views. Under this framework, five features, i.e., spatiotemporal interest points (STIP), dense trajectory

shape (DT-Shape), dense trajectory motion boundary histograms (DT-MBH), skeleton trajectory shape (ST-Shape), and skeleton trajectory motion boundary histograms (ST-MBH), are extracted from each view, separately. Then the three views are combined to derive a complete description of the 3D data. The first three features are representative for actions in intensity data but adapted to depth sequences. The last two are proposed by us, termed as skeleton-based features unique for 3D depth data. The RGB-D sensors, e.g., the Kinect, provide 3D positions of 20 skeleton joints and the evolution of each skeleton joint over time corresponds to one skeleton trajectory. Features aligned with the skeleton trajectory include shape descriptor (ST-Shape) and motion boundary histograms (ST-MBH), are extracted to characterize the actions with sparse trajectories. The five features characterize action patterns from different aspects, among which the top three best features are selected and fused based on a probabilistic fusion approach (PFA). We evaluate the proposed framework on three challenging depth action datasets. The experimental results show that the proposed TriViews framework achieves the most accurate results for depth-based action recognition, better than the state-of-the-art methods on all three databases.

Compared to human actions, animal behaviors exhibit some different characteristics. For example, animal body is much less expressive than human body, so some visual features and frameworks which are widely used for human action representation, cannot work well for animals. We investigate two features for mice behavior recognition, i.e., sparse and dense trajectory features. Sparse trajectory feature relies on tracking heavily. If tracking fails, the performance of sparse trajectory feature may deteriorate. In contrast, dense trajectory features are much more robust without relying on the tracking, thus a fusion approach is proposed for the integration of these two features. Experimental results on two public databases show that the integration of sparse and dense trajectory features can improve the recognition performance. Furthermore, the proposed approach outperforms the state-of-the-art methods on both databases.

Dedicated to my family.

Acknowledgements

First of all, I would like to express my sincere thanks to my advisor, Dr Guodong Guo for his guidance, encouragement and help during my graduate study as a research assistant. I appreciate his support and trust in me.

Second, I'd like to thank my committee members: Dr Cun-Quan Zhang and Dr Donald Adjeroh and all my other friends in Morgantown, West Virginia, for their help during the completion of my thesis.

I would also like to thank the Lane Department of Computer Science and Electrical Engineering at West Virginia University for providing me with a great study environment during my years as a graduate student.

Finally, I give my thanks to my family, my parents and sister. Without their continuous support, encourage and love, this would not have been possible.

Contents

1

Introduction

1.1 Motivation

Automatic human action recognition and understanding is an important yet challenging research topic. It has a wide range of real-world applications, such as human computer interaction (HCI), video surveillance, video retrieval, health care, etc. [1, 42, 53, 62]. In the past decades, research on action recognition has been mainly focused on using RGB or gray level intensity videos. Very recently, with the emergence of low-cost RGB-D sensors, e.g., the Kinect, human action recognition in 3D data has received great attentions. Compared to traditional color images/videos, depth maps/sequences exhibit several advantages. For example, the depth maps provide the 3D geometry and shape cues, which offer more discerning information than 2D videos to recover postures and recognize actions. Second, depth images/videos are insensitive to illumination changes. Depth cameras or RGB-D sensors can work in total darkness, which can benefit many applications such as some monitoring systems in lightless environment. Third, 3D human skeleton joints positions can be estimated from the depth data [46] with a reasonably good accuracy. Therefore, it is interesting to study how to fully utilize the 3D data for action recognition.

Compared to human action recognition, animal behavior analysis is not well-studied yet, although it is of great importance in practice. For instance, animal behavior analysis plays an important role in areas such as neuroscience, genetics, and pharmacology [18, 33, 37, 51].

1.2 Related Work on Spatiotemporal Interest Points

Spatiotemporal interest points (STIP) are typically localized at spatiotemporal key points where a sudden change occurs in both space and time. Local STIP features can be extracted around the interest points, which offer some robustness to clutter, occlusion, and intra-class variations.

STIP features have been shown successful for action recognition in RGB videos [45, 59], there are also some works exploring the STIP features for depth-based action recognition. Zhu *et al.* [71] transformed depth data into gray level depth videos, adapted STIP features from RGB to depth videos. Zhao *et al.* [70] proposed to adapt STIP detected on RGB videos to depth sequences and combine the two channels for action recognition. Ni *et al.* [38] proposed to add depth information to HOG/HOF descriptor. They divided the video into different depth layers and formed a multi-channel STIP histogram. Zhang and Parker [69] proposed another approach to utilize the depth information. They viewed depth as an additional dimension and extended the 3D Cuboids feature [12] to the fourth dimension. Xia and Aggarwal [64] proposed a new descriptor called Depth Cuboid Similarity Feature (DCSF) for depth action recognition. DCSF is a self-similarity feature, which computes a histogram of depth pixels in 3D blocks. Zhu *et al.* [72] evaluated the STIP features for depth-based action recognition. However, these works do not project the STIP features into three views as we do.

1.3 Related Work on Trajectories

Trajectories are extracted from the temporal tracking of spatial points. Messing *et al.* [36] tracked a set of keypoints in video sequence with the KLT tracker [35]. Trajectories were then represented as sequences of quantized velocity in log-polar coordinates. For action recognition, they employed a generative mixture model to combine the trajectories. Sun *et al.* [49] got trajectories by matching SIFT points over consecutive frames. Actions were characterized with three levels of contexts: point level, intra-trajectory level, and inter-trajectory level. Raptis and Soatto [43] tracked feature points in regions of interest. They proposed tracklet descriptors along the trajectories to encode the motion information. A recent work by Wang *et al.* [57] used dense trajectories to characterize video information. In their work, the trajectory shape, histograms of

oriented gradients (HOG), histograms of optical flow (HOF) and motion boundary histograms (MBH) are computed to encode the trajectory shape, local motion and appearance information. For action classification, they used the SVM classifier. Jiang *et al.* [21] computed dense trajectory for local patches. To model motion information such as object relationships, they used global and local reference points. However, the above works focus on intensity videos rather than 3D depth videos. Furthermore, there is no projection into three views as ours. Also, there is no study on using dense trajectory features for animal behavior recognition yet.

1.4 Related Work on Depth-based Human Action Recognition

A number of algorithms and features have been proposed for 3D depth-based action recognition. According to the data sources, these approaches may be divided into three categories: depth maps, 3D skeleton joints positions, and multiple data modalities.

In the first category, Li *et al.* [32] proposed to characterize a set of salient postures with a bag of sampled 3D points. In order to get the representative 3D points, they projected the depth map onto three orthogonal Cartesian planes and sampled a specified number of points at equal distance along the contours of the projections. They employed action graph to model the dynamics of actions. Vieira *et al.* [54] proposed the space-time occupancy patterns (STOP) to encode depth sequences. They divided both space and time axes into multiple segments, thus each depth sequence was divided into multiple 4D grids. Occupancy feature is computed in each grid by using the number of points within the grid. A nearest neighbor classifier was applied for recognition. Similarly, Wang *et al.* [60] also divided the depth sequence into multiple 4D volumes. They computed a semi-local feature called Random Occupancy Patterns (ROP) in each cell. Sparse coding was utilized to encode the features and the SVMs were used for classification. Yang *et al.* [68] proposed a different approach based on Motion Energy Images (MEI). They projected the depth map onto three orthogonal Cartesian planes and computed histogram of oriented gradients (HOG) from depth motion map (DMM) for each projection. Then they combined the HOG features of all the three projections to represent human actions. More recently, Oreifej and Liu [40] characterized the depth data as a surface in the 4D space of time, depth, and spatial

coordinates. Histograms of the 4D oriented surface normals (HON4D) were used for action recognition. To quantize the surface normals, they employed a 600-cell polychorons in the 4D space. For classification, they used the SVMs classifier. In [64], a modified spatiotemporal feature based on Cuboids was proposed to capture the action motion. They also used a correction function to suppress the flip noise in depth videos. A feature selection scheme based on the F-score was applied to the proposed features. The selected features were fed into the SVMs for action classification.

In the second category, skeleton joints positions are used to encode shape and motion information. Xia *et al.* [65] built a coordinate system based on the skeleton joints and divided the 3D space into different bins. By counting the number of joints in each bin, histogram of 3D skeleton joint locations (HOJ3D) was obtained as a compact representation of postures. The K-means clustering was used to construct the vocabulary and discrete Hidden Markov Model was used for action classification. Miranda *et al.* [31] proposed a pose descriptor in a torso-based coordinate system and used the SVM classifier to learn key poses. A decision forest was then employed to recognize the actions. Theodorakopoulos *et al.* [52] also computed the skeleton feature in a torso-based coordinate system. In order to obtain robust and invariant pose representations, they transformed the feature to a new domain called dissimilarity space [41]. Yang and Tian [66] proposed another skeleton feature called EigenJoints to model the human action posture. EigenJoints are a combination of static posture feature, motion feature, and the offset feature. They employed Naive-Bayes-Nearest-Neighbor (NBNN) for action classification. More recently, Devanne *et al.* [11] proposed to represent human actions with spatiotemporal motion trajectories. They concatenated the 3D positions of 20 skeleton joints into a 60-dimensional vector and viewed the evolution of such a vector over time as a trajectory. To better characterize the trajectory shape, they also employed the Riemannian manifold learning method [22].

In the third category, more than one data source is used. Wang *et al.* [61] proposed to combine the pairwise skeleton feature and local occupancy feature (LOP) and employed Fourier temporal pyramid to encode the temporal dynamics of the actions. Sung *et al.* [50] combined all the three channels, i.e., RGB, depth and skeleton joints positions, for action recognition. Hand positions, body pose and motion features were extracted from skeleton joints. HOG was used as the descriptor for both RGB and

depth images. A two-layer maximum-entropy Markov model was trained for classification. More recently, Zhu *et al.* [71] proposed to fuse STIP features extracted from depth sequence and skeleton joints feature with Random Forests method. A high recognition accuracy is achieved based on the fusion.

1.5 Related Work on Mice Behavior Understanding

Recently, mice behavior recognition has attracted some attentions to computer vision researchers. Dollar et al. [12] proposed to use a sparse spatiotemporal feature called Cuboids for a single mouse behavior recognition. Belongie et al. [6] computed Cuboids feature to capture caged mice's local motion. For classification, they employed linear discriminant analysis. Jhuang et al. [19] proposed to compute space-time motion features as well as some position and velocity-based features to characterize animal behaviors in a home-cage. For behavior classification, they employed the Hidden Markov Model Support Vector Machine [3]. These three works were to recognize actions of a single mouse.

More recently, Burgos-Artizzu et al. [9] tried to study social behaviors between two mice. They proposed trajectory features (TF) to characterize the location and motion information of mice. They also computed some widely used spatiotemporal interest point features (STIP) and combined TF with STIP feature for social behavior recognition in mice. For classification, they used the AdaBoost [15]. Eyjolfsdottir et al. [13] used the same trajectory feature as [9], but they proposed another classifier called Structured SVM for animal behavior recognition. Giancardo et al. [16] extended the social behavior recognition work from two mice to three mice. They proposed some spatiotemporal features to characterize position, motion and temporal information of mice. The spatiotemporal features were fed into the Random Forest classifier for behavior recognition. Also note that different from others using color camera to monitor the animals, [16] used an infrared camera to record mice behaviors.

On the other hand, dense trajectory features have shown very good performance for human action recognition. Wang et al. [57] proposed dense trajectories to encode video information for human action recognition. The trajectory shape, histograms of gradients (HOG), histograms of optical flow (HOF) and motion boundary histograms (MBH) were computed. Jiang et al. [21] proposed to cluster dense trajectories, and

use the cluster centers as motion reference points, thus the object relationships can be modeled. Vig et al. [55] proposed a saliency-based pruning stage to prune background features, which results in a more compact representation. More recently, Wang et al. [58] improved dense trajectories by estimating camera motions. These studies motivated us to explore dense trajectory features for mice behavior analysis, which has not been studied before.

Trajectory features (TF) computed from the tracked positions have been shown promising in mice behavior recognition [9, 13, 16], however, an animal track is composed of only a few points (only one or three points being tracked), it can easily become incorrect or erroneous. In contrast, dense trajectory features can overcome this problem since they are computed based on densely sampled feature points directly. When the tracking fails, dense trajectory features might correct the errors caused by tracking. In our approach, we study these two different features separately, and examine their differences. We also fuse the sparse trajectory features and dense trajectory features, using different fusion methods, to see if any improvement can be achieved.

1.6 Contributions

First, we conduct an evaluation of spatiotemporal interest points on four depth databases and find out the best detector and descriptor combination for each database. Second, we present a framework called TriViews for RGB-D data processing. Third, we investigate five kinds of features for depth-based action recognition, including two new features: skeleton trajectory shape (ST-Shape) and skeleton trajectory motion boundary histograms (ST-MBH). Fourth, we investigate and compare two kinds of features for mice behavior recognition: sparse and dense trajectory features. Moreover, we propose to combine the two features in a decision-level fusion scheme.

The remainder of the thesis is organized as follows: We first conduct the evaluation on four depth databases in Chapter 2. Then we present the TriViews framework for human action recognition with RGB-D data in Chapter 3. In Chapter 4 we present our work for mice behavior recognition & analysis. Finally, we conclude in Chapter 5 and discuss possible future works.

2

Evaluating Spatiotemporal Interest Point Features for Depth-based Action Recognition

2.1 Overview of The Method

STIP features have been shown promising performances for human action recognition in RGB videos, Kinect provides depth data, which is a new modality for human action recognition. In this chapter, we evaluate the spatiotemporal interest point (STIP) based features for depth-based action recognition. Figure 2.1 shows the framework of the proposed approach. Different interest point detectors and descriptors are first combined to form various STIP features. Then the bag-of-words representation and the SVM classifiers are used for action learning. Our comprehensive evaluation is conducted on four challenging 3D depth databases. Further, we use two schemes to refine the STIP features, one is to detect the interest points in RGB videos and apply to the aligned depth sequences, and the other is to use the human skeleton to remove irrelevant interest points. These refinements can help us have a deeper understanding of the STIP features on 3D depth data. Finally, we investigate a fusion of the best STIP features with the prevalent skeleton features, to present a complementary use of the STIP features for action recognition on 3D data. The fusion approach gives significantly higher accuracies than many state-of-the-art results.

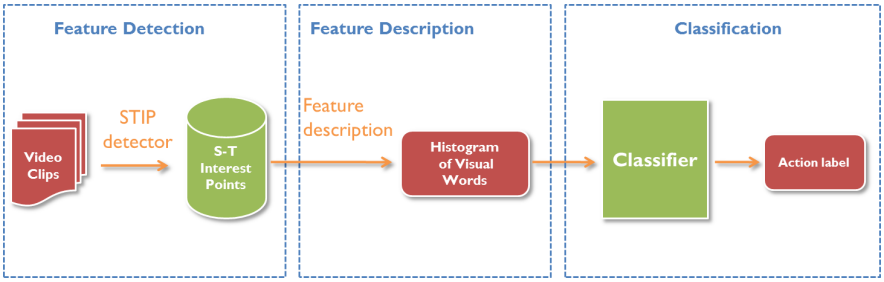

Figure 2.1: Framework of the proposed method for depth-based human action recognition.

2.2 Spatiotemporal Interest Point Features

Different Spatiotemporal Interest Point (STIP) features have been proposed for action characterization in RGB videos with good performance [59]. For example, Laptev and Lindeberg [29] used some effective methods to make STIP velocity-adaptive as well as spatially and temporally invariant. Willems *et al.* [63] presented a method to detect features under scale changes, in-plane rotations, video compression and camera motion, the extended SURF descriptor was also proposed in this work. Dollar *et al.* [12] proposed the cuboids detectors and descriptors for action analysis. Jhuang *et al.* [20] used local descriptors with space-time gradients as well as optical flow. Klaser *et al.* [24] compared space-time HOG3D descriptor with HOG and HOF descriptors [30]. Recently, Wang *et al.* [59] conducted an evaluation of different detectors and descriptors on four RGB/intensity action databases. Shabani *et al.* [45] evaluated the motion-based and structured-based detectors for action recognition in color/intensity videos. However, there is no systematic evaluation of the STIP features on 3D depth videos.

In Wang *et al.*'s work [59], it was observed that although the spatiotemporal interest point features perform differently on different databases, their performances are quite similar on the same database. Our evaluation will show that the STIP features perform quite differently on the same depth database (See Section 2.4). In the following, we introduce the specific STIP features that are used in our evaluation.

2.2.1 Interest points detectors

The Harris3D detector was proposed in [27]. It locates the spatiotemporal volumes with large variations along space and temporal directions in a video sequence. A spatiotemporal second-moment matrix is used to model a video sequence f, $\mu = g(\cdot) \times \begin{pmatrix} L_x^2 & L_x L_y & L_x L_t \\ L_x L_y & L_y^2 & L_y L_t \\ L_x L_t & L_y L_t & L_t^2 \end{pmatrix}$, where $g(\cdot)$ is a Gaussian function for weighting and L is the convolution of f with a spatiotemporal Gaussian derivative kernel. The interest point locations are determined by computing the local maxima of the response function $H = det(\mu) - k \cdot trace^3(\mu)$.

The Cuboids [12] detector computes the interest point location by the local maxima of the response function R, which is defined as: $R = (I * g * h_{ev})^2 + (I * g * h_{od})^2$, where g is the 2D Gaussian smoothing kernel, h_{ev} and h_{od} are a quadrature pair of 1D Gabor filter, which are computed by $h_{ev} = -cos(2\pi t\omega)e^{-t^2/\tau^2}$ and $h_{ev} = -sin(2\pi t\omega)e^{-t^2/\tau^2}$.

Willems *et al.* [63] proposed the Hessian detector, which measures the strength of each interest point using the Hessian matrix. The response function is defined as $S = |det(H)|$, where H is the Hessian matrix.

2.2.2 Local feature descriptors

Given a set of interest point locations, various feature descriptors can be applied to characterize the local space-time content. Given the spatial scale σ and temporal scale τ at each interest point location, a local volume is used to extract features.

Kläser *et al.* extended the histograms of oriented gradient (HOG) to HOG3D, which is the histogram of 3D gradient orientations. Integral videos are computed for efficiency.

HOG/HOF descriptor was proposed by Laptev *et al.* [30], using the combination of histogram of gradient (HoG) and histogram of optical flow (HoF) accumulated from the local volume.

The Cuboids descriptor was proposed along with the Cuboids detector in [12]. For each detected point (x, y, t, σ, τ), a feature descriptor is computed in a 3D patch centered at (x, y, t). The gradient at each spatiotemporal location is computed within the cuboid and the histogram is computed as the feature vector. The PCA can be applied to reduce the dimensionality.

The extended SURF (ESURF) descriptor [63] was proposed with the Hessian detector, which is an extension of the SURF [5]. For each local volume, the feature vector is computed using the sum of uniformly sampled responses of Haar-waveletes along three directions.

We will evaluate the above three interest point detectors and six local descriptors for 3D action recognition. Although there exist some works using the STIP features for depth-based action recognition [64, 69, 70], only very limited types of STIP features were investigated. Through the evaluation of several representative STIP features on multiple depth databases, we will not only provide the benchmark results of STIP features on depth data, but also find the best, appropriate STIP features that may help to improve the accuracies significantly [71] for depth-based action recognition.

2.3 Databases

Table 2.1: Depth-based action/activities databases. In the 4th column, RGB denotes color images, DEP denotes depth maps, and SK denotes skeleton joints positions. The 5th column shows the average length of each video in the dataset.

Database	# of Actions	# of Subjects	# of sequences	# of channels	Video Length
MSR-Action3D	20	10	557	DEP, SK	˜1s
MSRDailyActivity3D	16	10	320	RGB, DEP,SK	˜6s
UTKinect-Action	10	10	200	RGB, DEP,SK	˜3s
CAD-60	12	4	60	RGB, DEP,SK	˜45s

In order to perform a comprehensive evaluation, we conduct experiments on four different depth databases, which were captured under different scenarios and/or environments. The evaluation on these databases can provide a thorough test of various STIP features on depth data. Table 2.1 shows a brief description of the four depth-based action/activity databases. More details of these databases are given as follows.

2.3.1 MSR-Action3D Dataset

MSR-Action3D Dataset [32] was captured by a depth camera similar to the Kinect sensor. This dataset contains 20 actions, and each action was performed by 10 subjects three times. Two channels of data are provided: depth sequences at 15 frames per second (fps) with resolution of 640×480, and skeleton joint positions in each frame. The 20 actions are: *high arm wave, horizontal arm wave, hammer, hand catch, forward*

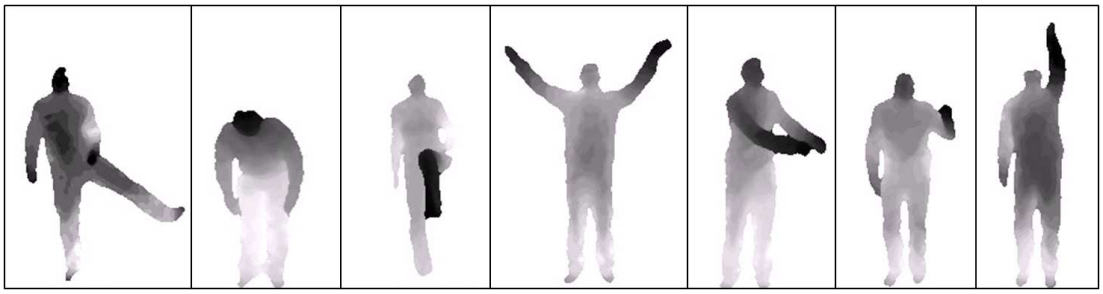

Figure 2.2: Some samples from MSRAction3D Dataset. 7 depth images are showed. The actions shown are (from left to right): side kick, bend, jog, high arm wave, golf swing, pickup&throw and high throw.

Table 2.2: Three subsets of actions used for the experiments on MSRAction3D dataset.

AS1	**AS2**	**AS3**
Horizontal arm wave	High arm wave	Hight throw
Hammer	Hand catch	Forward kick
Forward punch	Draw x	Side kick
High throw	Draw tick	Jogging
Hand clap	Draw circle	Tennis swing
Bend	Two hand wave	Tennis serve
Tennis serve	Forward kick	Golf swing
Pickup & throw	Side boxing	Pickup & throw

punch, high throw, draw x, draw tick, draw circle, hand clap, two hand wave, sideboxing, bend, forward kick, side kick, jogging, tennis swing, tennis serve, golf swing, and *pick up & throw* (see Figure 2.2 for some example images).

2.3.2 MSRDailyActivity3D Dataset

This dataset was collected for human daily activities by a Kinect device [61]. In total there are 16 activities in this dataset: *drink, eat, read book, call cellphone, write on a paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lay down on sofa, walk, play guitar, stand up,* and *sit down.* Each subject performed an activity twice, one "sitting on sofa" and the other "standing". The total number of videos is 320. Three channels of data, i.e., RGB, depth and skeleton joint positions are provided in this dataset. See Figure 2.3 for some examples of depth images in this

Figure 2.3: Sample depth images from MSRDailyActivity3D Dataset. Actions in the top row (left to right): use laptop, use vacuum cleaner, cheer up, and lay down on sofa. Action classes in the bottom row: toss paper, stand up, walk, and play guitar.

Figure 2.4: Sample images from UTKinect-Action Dataset. Action classes in the top row: walk, wave hands, sit down, and throw. Action classes in the bottom row: pick up, clap hands, carry and push.

dataset.

2.3.3 UTKinect-Action Dataset

The action videos of the UTKinect-Action Dataset [65], were collected by a single stationary Kinect with the distance ranges from 4 to 11 feet. There are totally 10 action classes performed by 10 subjects. Each subject performed each action twice. The RGB, depth and skeleton joint locations are synchronized and all three channels are provided. Some examples of depth images are shown in Figure 2.4. The resolution of RGB images is 640×480, the depth image resolution is 320×240. The 10 action classes are: *walk, sit down, stand up, pick up, carry, throw, push, pull, wave hands,* and *clap hands.*

Figure 2.5: Examples depth images from the CAD-60 Dataset to illustrate the actions.

2.3.4 CAD-60 Dataset

Cornell Activity Dataset-60 (CAD-60) [50], contains 60 RGB-D videos collected by a Kinect sensor with the distance ranges from 1.2m to 3.5m, the resolution of the depth sequences is 640×480, and captured at 15 fps. There are 4 different subjects and 12 different actions. The action videos were captured in five different locations, with 3 to 4 common activities performed at each location. The five locations are: office, kitchen, bedroom, bathroom and living room. Figure 2.5 shows some example depth images from this dataset. All the RGB, depth and skeleton data are provided in this dataset.

2.4 Evaluations

We present the experimental settings in Section 2.4.1, the evaluation results for various combinations of detectors and descriptors in Section 2.4.2, and two STIP refinement approaches along with the corresponding results in Section 2.4.3.

2.4.1 Experimental settings

The bag-of-words representation is used for the spatiotemporal interest points. First, different STIP detectors are applied to the depth sequences. Given the detected locations, different local descriptors are used to characterize the space-time volume around each interest point. These local features are then quantized into visual words, so that a depth action sequence can be represented as a histogram of the visual words. In our evaluation, vocabularies are constructed using the K-means clustering technique. We empirically set the vocabulary size to be 200, 300, 850 and 1550, respectively, for the MSRDailyActivity3D, MSRAction3D, CAD-60 dataset and UTKinect-Action datasets, depending on the database size and empirical performance. After quantization, the histograms of visual words are used as the features for action classification. The multi-class support vector machines (SVMs) are used for action learning, with a linear kernel for the CAD-60 dataset and χ^2-kernel for the other three datasets, based on our empirical comparisons between different kernels. The χ^2-kernel is defined by: $K(H_i, H_j) = exp\left(-\frac{1}{2A}\Sigma_{n=1}^{V}\frac{(h_{in}-h_{jn})^2}{h_{in}+h_{jn}}\right)$, where $H_i = \{h_{in}\}$ and $H_j = \{h_{jn}\}$ are the frequency histograms of the visual word occurrences, and V is the vocabulary size. A is the mean value of distances between all training samples.

For different feature representations, we utilize the implementations or source code provided by the authors, mostly with the default parameter settings, since some executable code cannot be modified. All the experiments were conducted on a 64-bit operating system DELL Optiplex 790 PC, with i7 3.4GHz CPU and 12G RAM.

Specifically, for the Harris3D detector, we used the original implementation with the default parameter settings: $k = 0.0005, \sigma^2 = [4, 8, 16, 32, 64, 128]$ and $\tau^2 = [2, 4]$. For the Cuboids detector [12], we ran the authors' implementation and the default scale values $\sigma = 2, \tau = 4$ were used in our evaluation. The UTKinect-Action dataset has typically shorter video clips, we used $\sigma = 2, \tau = 2$ for the Cuboids detector. For the Hessian detector [63], the executable code was used with the default parameter setting.

For the HOG/HOF descriptor, we followed [30] and adopted the grid parameters $n_x = n_y = 3, n_t = 2, \sigma^2 = 4$ and $\tau^2 = 2$. For the HOG3D descriptor [24], we used the parameters $n_x = n_y = 5, n_t = 4, \sigma = 2$ and $\tau = 2$ for the UTKinect-Action dataset and $n_x = n_y = 2, n_t = 5, \sigma = 2$ and $\tau = 4$ for the other three datasets in our evaluation. For the Cuboid descriptor [12], we applied the descriptor size $\Delta_x(\sigma) = \Delta_y(\sigma) = 2\sigma + 1, \Delta_t(\tau) = 2\tau + 1$, where $\sigma = 2$, $\tau = 4$. The PCA was applied to reduce the feature dimensions to 100. For the ESURF descriptor, we used the executable code with default parameter settings [63]: $\Delta_x(\sigma) = \Delta_y(\sigma) = 3\sigma, \Delta_t(\tau) = 3\tau$.

For all depth databases, the depth sequences are firstly transformed and stored into gray level videos (depth videos). The skeleton joint positions are also stored for each frame. Then the spatiotemporal features are extracted from the depth videos for each database.

2.4.2 Evaluation Results

The evaluation results are presented in the following, using all four datasets.

2.4.2.1 On MSRAction3D Dataset

MSRAction3D is a commonly used dataset for 3D action recognition. We followed the same settings as [32], where the dataset is divided into 3 subsets, each consisting of 8 actions (see Table2.2). Then a cross-subject scheme is used in our evaluation, with half of the subjects for training and the remaining half for testing. The overall accuracy is computed by taking the average over the three subsets. The results of different detectors/descriptors on this dataset are showed in Table 2.3. One can see that the STIP features have very different accuracies on the same database, ranging from 47.1% to 80.8%, when different detectors and descriptors are used. This observation is very different from the results on color/gray level action videos [59], where the different STIP features have similar accuracies on the same database. This evaluation indicates the significant difference between 3D depth and color/gray level videos in action recognition.

The highest accuracy is achieved by Harris3D+HOG/HOF feature with a recognition accuracy of 80.8%. This accuracy is comparable to some state-of-the-art approaches, but lower than the highest in the literature by more than 10% (see Table 2.10 for the state-of-the-art results on MSRAction3D). Note that in [61] the skeleton joints information was used while in our evaluation of STIP features, only the depth

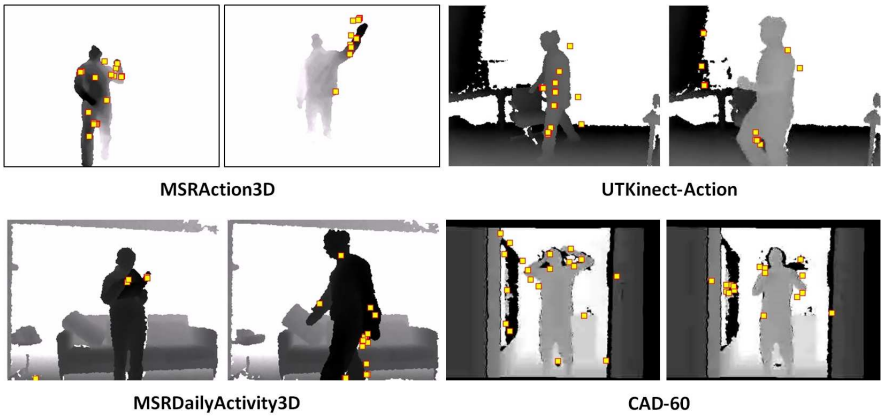

Figure 2.6: Illustration of the spatiotemporal interest points detected on depth sequences from four datasets.

Table 2.3: Accuracies of different STIP features on MSRAction3D dataset. Different detectors and descriptors are combined. Some combinations cannot be realized because of the non-separable executable code.

	HOG3D	**HOG/HOF**	**HOG**	**HOF**	**Cuboids**	**ESURF**
Harris3D	76.1%	**80.8%**	72.3%	77.3%	-	-
Cuboids	77.3%	78.7%	68.5%	71.0%	70.0%	-
Hessian	60.3%	55.9%	47.3%	44.9%	-	47.1%

videos are used. One reason that might impact the accuracy is that the interest points cannot be detected for several depth sequences where the lengths of the sequences are quite short.

2.4.2.2 On MSRDailyActivity3D Dataset

The MSRDailyActivity3D dataset contains 16 activities performed by 10 subjects in two scenarios: sitting and standing. Similar to the partition in [32], we divided this dataset into 2 subsets, and evaluate the performance considering two different scenarios, sitting and standing, respectively. We consider the activities in each subset according to the motions: subset 1 (AS1) contains activities without much motion and subset 2 (AS2) with obvious motion. Table 2.4 shows how we divide the subsets. In our evaluation, we adopt the cross-subject test scheme, using half of the subjects for training and the remaining half for testing. The final results are obtained by averaging accuracies over

Table 2.4: Subsets of actions used for the experiments on MSRDailyActivity3D dataset.

AS1	**AS2**
Read book	Drink
Write on a paper	Eat
Use laptop	Call cellphone
Use vacuum cleaner	Cheer up
Sit still	Lay down on sofa
Toss paper	Walk
Play game	Stand up
Play guitar	Sit down

Table 2.5: Accuracies of various STIP features on MSRDailyActivity3D dataset.

	HOG3D	**HOG/HOF**	**HOG**	**HOF**	**Cuboids**	**ESURF**
Harris3D	60.6%	67.5%	63.8%	59.4%	-	-
Cuboids	68.8%	**70.6%**	68.1%	58.1%	64.4%	-
Hessian	**70.6%**	63.8%	61.9%	63.1%	-	65.6%

the subsets.

The evaluation results on MSRDailyActivity3D dataset using different combinations of detectors and descriptors are presented in Table 2.5. Again, the STIP features achieved very different accuracies. The highest accuracy is obtained by Cuboids+HOG/HOF and Hessian+HOG3D, with an accuracy of 70.6%. The result is lower than the reported results, e.g., Oreifej *et al.* got 80% accuracy with HON4D feature in [40]. The highest accuracy from previous approaches is 85.8% obtained in [61]. In our evaluation, all the combinations of detector/descriptors are above 58%. In the subset with more motion, the performance of STIP is much better ($\sim$ 80%) than the subset with less motion ($\sim$ 50%). This demonstrates that the STIP features can characterize actions with significant motions, but not static actions like sitting. Further, the STIP features cannot represent the human-object interaction. There are several activities in this dataset with similar motion but different objects, e.g., reading and writing, eating and drinking, etc. We also observe that many of the interest points are detected on depth sequences irrelevant to the actions (see Figure 2.7). This inspires us to evaluate some refinement schemes for the STIP features (to be shown later).

Figure 2.7: Examples of interest points that are detected from the background (MSRActivity3D dataset).

Table 2.6: Accuracies of various STIP features on UTKinect-Action dataset. Note that we use half subjects for training and the remaining half for testing. There are 100 samples in total in the test set.

	HOG3D	**HOG/HOF**	**HOG**	**HOF**	**Cuboids**	**ESURF**
Harris3D	**81.0%**	80.0%	66.0%	69.0%	-	-
Cuboids	65.0%	65.0%	56.0%	57.0%	67.0%	-
Hessian	69.0%	56.0%	57.0%	53.0%	-	65.0%

2.4.2.3 On UTKinect-Action Dataset

The evaluation results on the UTKinect-Action dataset are showed in Table 2.6. Note that because many depth sequences in this dataset are of length about 10 frames, which is too short for space-time interest point detection. Thus a preprocessing is conducted for the depth videos where 10 frames are copied to expand the length of video from both the starting and ending frames.

From the results, the best accuracy is 81%, obtained by Harris3D+HOG3D. This result is lower than the result 90.9% in [65], and the highest accuracy 91.5% in [11]. Note that in [65] and [11] the *leave-one-out cross-validation* scheme was applied but

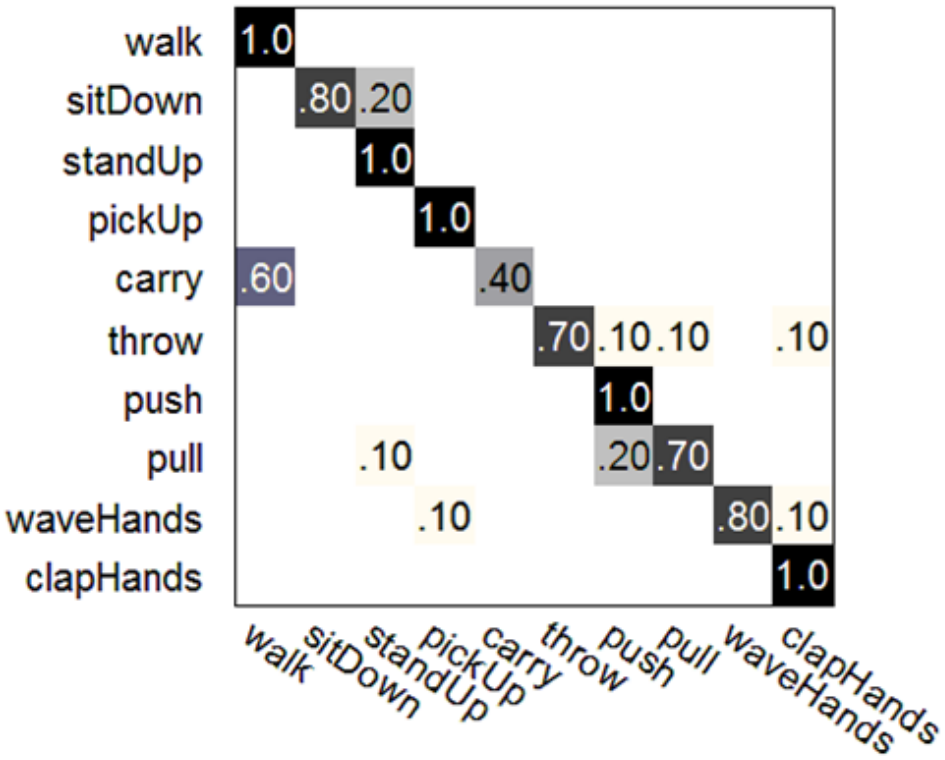

Figure 2.8: Confusion matrix for the feature Harris3D+HOG3D on UTKinect-Action dataset.

we use half of the subjects for training and the other half for testing. Figure 2.8 shows the confusion matrix of the best STIP feature. Most of the actions are correctly recognized, while the action "carry" has a much lower recognition rate, i.e., 60% of the testing samples are incorrectly classified as "walk". These two actions are quite similar in the dataset, since "carrying" is performed by a "walking" subject who holds an object. The STIP features might mainly focus on the body motions rather than a relatively small object.

2.4.2.4 On Cornell Activity Dataset (CAD-60)

For the CAD-60 dataset, all the depth videos are sampled to 500 frames in our evaluation. All the activity categories (12 desired activities and a random activity) in

Table 2.7: Accuracies of various STIP features on CAD-60 dataset.

	HOG3D	**HOG/HOF**	**HOG**	**HOF**	**Cuboids**	**ESURF**
Harris3D	43.8%	50.0%	43.8%	37.5%	-	-
Cuboids	50.0%	31.3%	37.5%	37.5%	43.8%	-
Hessian	43.8%	50.0%	56.3%	43.8%	-	**62.5%**

this dataset are used in our evaluation as in [50]. The same experimental settings are adopted, i.e., three subjects for training, while the remaining for testing.

The evaluation results are shown in Table 2.7. Among the various features, the Hessian+ESURF gives the highest accuracy 62.5%. From the confusion matrix (Figure 2.9), one can see that some of the similar activities on depth sequence are incorrectly recognized, e.g., talkOnCouch and relaxOnCouch, and the random activity in this dataset also influences the recognition rate, where the talkOnPhone activity is recognized incorrectly as the random activity.

In [50], the precision/recall is reported as the performance measurement (67.9%/55.5%). Yang *et al.* [67] reported 71.9%/66.6% on this dataset. Koppula *et al.* [25] reported the 80.8%/71.4%. We also compute precision/recall for the feature Hessian+ESURF. The result achieves 66.7%/59.0%. Note that in our experiment we do not divide the different environment into different subset as [25]. The noisy background in depth sequences (see Figure 2.6) impact the detection of interest points with many interest points detected from the background. This drawback can be overcome when human segmentation is applied. We will investigate some refinement to reduce the effect of background noise on depth-based action recognition.

2.4.3 Refinements of the STIP features

In the above experiments, various STIP features are evaluated on depth videos with recognition accuracies reported. The best accuracies on each database are comparable to, but lower than some state-of-the-art methods that are developed especially for 3D action analysis. Note that the synchronized RGB videos and the human skeleton joints positions [46] are usually provided with the depth sequences. Intuitively these different sources of data can be used as the complementary information for human action recognition. Thus in our evaluation, we attempt to further utilize the RGB videos

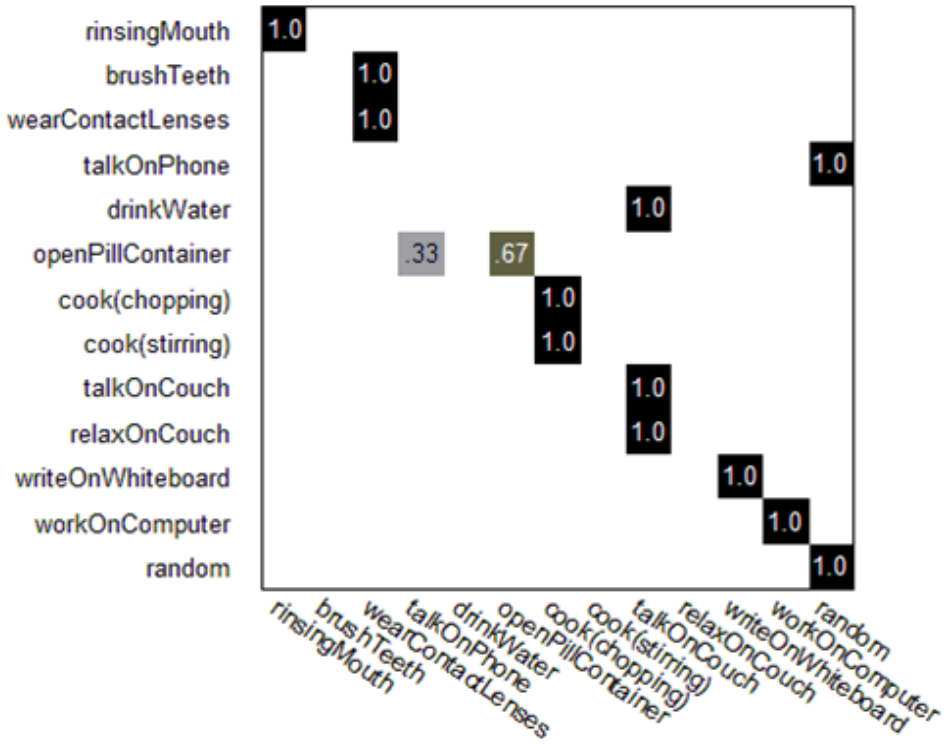

Figure 2.9: Confusion matrix for the feature Hessian+ESURF on CAD-60 dataset.

and the skeleton joints positions, to enhance the performance for action recognition on depth data. In this way, we can understand the STIP features deeper in depth videos. Two approaches are investigated in the following.

2.4.3.1 STIP feature refinement using Skeleton Joints

Shotton *et al.* [46] developed an efficient technique for human skeleton detection with 20 joint positions. Since the STIP features have a drawback, i.e., the spatial relations or distributions of the interest points cannot be utilized. From the above experiments, we observe that the detected interest points on depth images can be in the background or not accurate because of the noise in depth data. Therefore, we demonstrate that on depth images, the refinement of interest point detection could be done by using the skeleton. It is based on constraining the locations of STIP according to the skeleton joints. The idea is different from the work [64], but aims at the same goal—interest

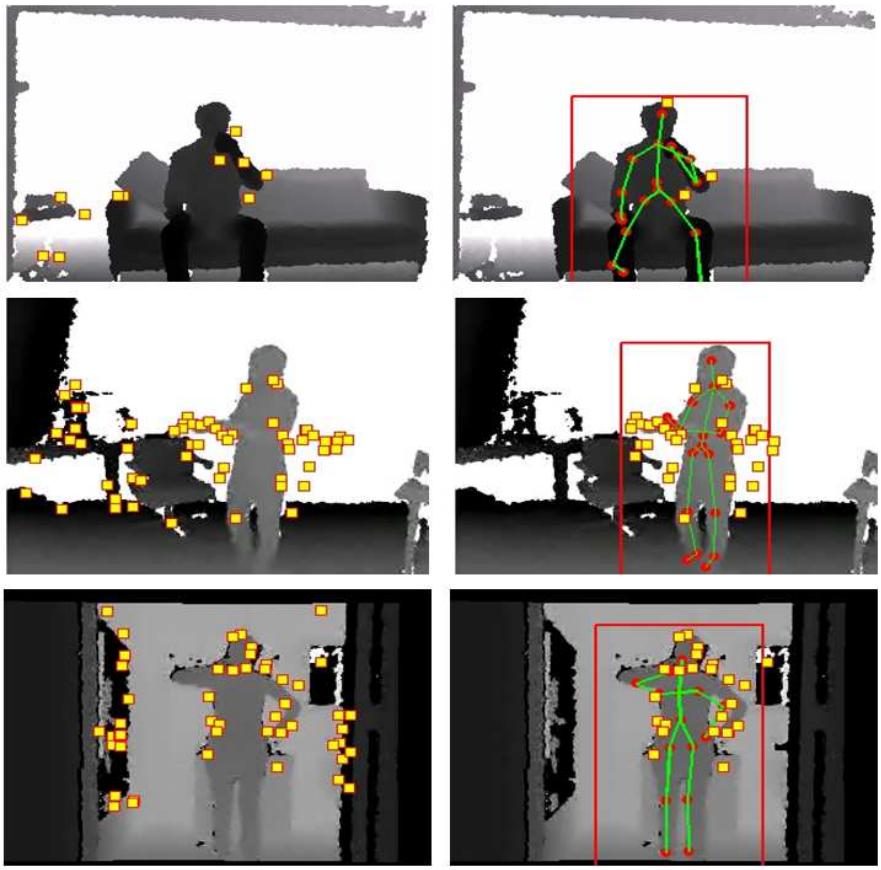

Figure 2.10: Examples of STIP refinement on different datasets. Left column shows the original interest points detected, right column shows the interest points after refinement by the human bounding box derived from the skeleton joints.

points refinement. Specifically, we define a bounding box around the subject at each frame t. The bounding box at frame t is obtained by the temporal images from time $t-5$ to $t+5$, and the maximum boundaries are selected and shifted by 30 pixels to each side to construct the new bounding box. Then the STIP which are detected on the whole depth sequences are constrained within the new box. STIP detections which lie outside the bounding box are considered as from the background, and thus are eliminated (see Figure 2.10). Finally, we do the evaluation again using the same experimental settings as previous, only a smaller K in K-means clustering because of the reduced number of interest points.

The evaluation results using this STIP refinement scheme on four datasets are

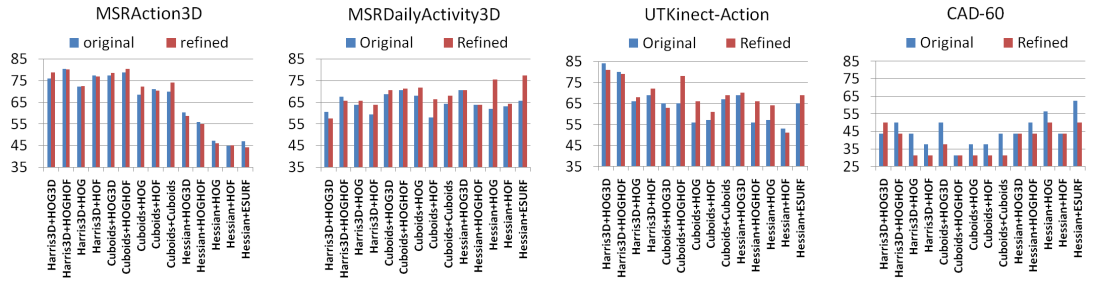

Figure 2.11: Bar graph of the recognition accuracies before and after the refinements on different datasets. The vertical axis denotes the recognition accuracy (%).

shown in Figure 2.11. From the results we observe that (1) most of the features can get better results when applied the STIP refinement, e.g., on MSRAction3D dataset, the accuracy of Cuboids + Cuboids feature increases by 4.2% after the refinement; on MSRDailyActivity3D dataset, an 11.9% increase is achieved for the Hessian + ESURF feature; and on UTKinect-Action dataset, the accuracy is increased by 13% for Cuboids + HOG/HOF feature. We also notice that on the CAD-60 dataset, the STIP refinement method does not improve the accuracies. One reason might be that the dataset was collected in five different locations and certain actions are "correlated" to some specific scene/location, e.g., the action 'cooking' is performed in kitchen, while the action 'brushing teeth' is performed in bathroom, etc. The eliminated STIPs, which are mainly from the background, could contain some helpful information for action encoding. Eliminating the interest points from background will "lose" the scene or context information, thus the refinement may have some negative impact on action analysis; (2) The overall accuracies on MSRAction3D and MSRActivity3D datasets increase after applying the STIP refinement. On MSRAction3D dataset, the refined accuracy is 80.5%, comparing to the original accuracy 78.7%, on MSRActivity3D dataset, the best accuracy is 77.5%, which is higher than the original 70.6%, after the refinement.

2.4.3.2 STIP feature refinement using RGB images and Skeleton Joints

We have shown above that in most cases the STIP refinement with the 20 skeleton joint positions can increase the action recognition rates. However, the performance is still highly relied on the interest point detection accuracy. When the interest point detection performs poorly on the depth maps because of the noisy depth data, the

Table 2.8: Accuracies using skeleton and RGB refinement approaches. Two cells have no results since the MSRAction 3D dataset does not contain RGB data.

	MSRDailyActivity3D	UTKinect-Action	CAD-60	MSRAction3D
Original	70.6%	81.0%	56.3%	80.8%
RGB Refined	75.6%	85.0%	68.8%	–
Skeleton Refined	72.5%	84.0%	50.0%	81.7%
Skeleton & RGB Refined	77.5%	85.0%	62.5%	–

skeleton constraints may not help too much. Based on this consideration, we pursue another refinement scheme. The idea is to adopt the interest point detection on RGB videos, i.e., using the STIP locations detected in RGB videos for depth sequences. In other words, the interest point detection is conducted on RGB sequences, and just duplicated to the depth maps. The feature descriptors are still executed on the depth videos.

Experiments are conducted on three datasets except the MSRAction3D because it does not have the RGB data. We use the same settings as previous. The evaluation results are shown in Table 2.8. The best STIP feature on each dataset are selected (because separate implementation of ESURF descriptor is not available, we chose the 2nd best STIP feature instead). From the results, one can see that the accuracies are improved significantly after using RGB refinement approach, either the skeleton refinement is applied or not. On MSRDailyActivity3D dataset, the accuracy is increased from 70.6% to 75.6%, on CAD-60 dataset, the accuracy is improved from 56.3% to 68.8%, and on the UTKinect-Action dataset, the accuracy is improved from 81.0% to 85.0%, when using the RGB refinement approach.

For the refinement with skeleton joints, the accuracies can be improved or keep the same on the MSRDailyActivity3D and UTKinect-Action datasets, but reduced on the CAD-60 dataset. The reason could be that the interest points located in the background or scene may help to improve the action recognition accuracies (the CAD-60 dataset contains different actions in different scenes), while the removal of those interest points (constrained by the skeleton joints) can reduce the recognition performance.

The refinement results show that it may not be accurate enough to use the detected locations of interest points on depth sequences directly, because of the noisy depth values.

2.5 Fusing spatiotemporal features and skeleton joints for action recognition

In the above, two approaches have been presented to refine the STIP features. These approaches can be viewed as posing constraints to the interest point locations on depth videos, by using either RGB videos or the skeleton joints. On the other hand, the skeleton joints positions extracted from the depth videos can be used as another feature, representing human posture information. In this section we want to evaluate the performance of combining the STIP features with the skeleton joints feature. This evaluation can tell if the STIP features can complement the skeleton joints features, and if the combination can improve the accuracies significantly. If the accuracies can be improved greatly, it can indicate the usefulness of the STIP features from another aspect.

Specifically, the combination approach has four major steps, which has been presented in a workshop [71]. Firstly, the STIP features are extracted on depth sequences. Then skeleton joints features are computed from the skeleton joint positions. A quantization is performed for the two features respectively to encode the action sequences with histograms. Finally, a feature-level fusion is executed for action recognition using the random forests method [7]. We chose the detector/descriptor combinations which performs the best based on our evaluation presented above. The evaluation of the STIP features in Section 2.4 is the basis for our fusion approach [71].

We use the histogram of the skeleton joints features proposed in [67] to combine with the best STIP features on each database. Different from [67] where the Naive Bayes classifier was used, we compute the histogram of the joints to combine with the STIP features by the random forests method.

The features from joint locations consist of three parts: (1) current posture: pairwise joint distances in current posture; (2) motion: joints difference between current posture and the original (in the first frame); and (3) offset: joints differences between current posture and the previous one. A concatenation of the three feature vectors is taken to represent the feature. The PCA method is applied for dimensionality reduction.

To represent each action sequence, we quantize the STIP features and the skeleton joints features, respectively, based on the K-means clustering. The cluster centers are

used as the keywords to construct the histogram bins. These features are used in the next step for feature-level fusion and action classification.

In order to perform the fusion and feature selection of spatiotemporal features and the skeleton joints features, the random forests (RFs) method [7] is used. RFs are usually considered as a classifier using tree predictors in which each tree splits the data depending on the randomly selected features. And there are many nice properties to use the random forests: (1) robustness to noise, (2) efficiency for classification, and (3) the improvement of accuracy by growing multiple trees and vote for the most popular class. Here we use the RFs for fusion of distinct features and action classification together.

The experiments are conducted on the four datasets (MSRAction3D, UTKinect-Action, CAD-60, and MSRDailyActivity3D) while three of them were used in our study in [71]. Our fusion approach can improve the recognition rates to 94.3%, 91.9%, 87.5%, and 80.0%, respectively, on the four databases, which are significantly higher than the STIP feature or skeleton. This result shows that the STIP features can be useful to complement the often-used skeleton features for action recognition.

We also compare the fusion results to other approaches reported in the literature on the four datasets. Table 2.10 shows all reported results that we can find on the MSRAction3D dataset. Under the same experimental settings, it can be seen that the fusion result of 94.3% accuracy is the second best result among all of the previous methods. Our result is only 0.5% lower than the best result in [39]. On the UTKinect-Action dataset from Table 2.11, the fusion approach has an accuracy of 91.9% which is higher than the DSTIP+DCSF feature [64], and slightly higher than the HOJ3D feature in [65] (90.9%) and the space-time pose representation in [11] (91.5%). Note that we used the same settings as [64], which is more challenging than the settings in [65] and [11]. On the CAD-60 dataset, the experimental settings are kept the same as [50] and the presicion/recall of our fusion method is computed for a direct comparison with other methods, shown in Table 2.12. Our fusion approach obtained a much higher accuracy than the state-of-the-art results on this dataset. Finally, Table 2.13 shows the results on the MSRDailyActivity3D dataset, an accuracy 80.0% is obtained using our fusion approach. Slightly different settings are used in our experiment, since the actions are divided into two groups to measure the performance difference between them. Our fusion result is comparable but about 8% lower than the highest accuracy. Note that

Table 2.9: Accuracies of the fusion method compared to each single feature on four datasets. RFs denotes the random forests method.

MSRAction3D	**Acc.**
STIP (Harris3D+HOG/HOF)	77.5%
Skeleton Joint Features	90.9%
Combined features with RFs	**94.3%**
UTKinect-Action	**Acc.**
STIP (Harris3D+HOG3D)	80.8%
Skeleton Joint Features	87.9%
Combined features with RFs	**91.9%**
CAD-60	**Acc.**
STIP (Hessian+ESURF)	75.0%
Skeleton Joint Features	81.3%
STIP + Skeleton	**87.5%**
MSRDailyActivity3D	**Acc.**
STIP (Hessian+HOGHOF)	70.6%
Skeleton Joint Features	73.8%
Combined features with RFs	**80.0%**

all the 16 activities are used in our experiment, while in [64], four activities (with less motion) were removed in their experiment.

From the comparison with various approaches, we demonstrate the usefulness of the STIP features for depth-based action recognition, when combined with the skeleton feature.

2.6 Conclusions

We have presented a comprehensive evaluation of the spatiotemporal interest point features for action recognition in 3D. The evaluated STIP features include three spatiotemporal interest point detectors and six descriptors. The combinations of these detectors and descriptors form 14 different features. These STIP features have been evaluated on four different depth action/activity databases. The comparisons to the state-of-the-art methods have shown that the STIP features are still useful for depth-

Table 2.10: Comparisons of different methods on MSRAction3D dataset.

Method	**Accuracy**
High Dimensional Convolutional Network [60]	72.5%
Action Graph on Bag of 3D Points [32]	74.7%
HOJ3D feature [65]	79.0%
Key Pose Learning [31]	80.3%
Eigenjoints [67]	82.3%
STOP feature [54]	84.8%
Random Occupancy Patterns [60]	86.2%
Actionlet [61]	88.2%
HON4D [40]	88.9%
DSTIP+DCSF [64]	89.3%
Depth Motion Maps [68]	91.6%
Space-time Pose Representation [11]	92.8%
JAS (Cosine)+MaxMin+HOG2 [39]	94.8%
STIP + Skeleton	94.3%

Table 2.11: Comparisons of different methods on UTKinect-Action dataset.

Method	**Accuracy**
DSTIP+DCSF [64]	85.8%
HOJ3D [65]	90.9%
space-time pose representation [11]	91.5%
STIP+Skeleton	**91.9%**

Table 2.12: Comparisons of different methods on CAD-60 dataset.

Method	**Precision/Recall**
J. Sung *et al.* [50]	67.9%/55.5%
X. Yang *et al.* [67]	71.9%/66.6%
Koppula *et al.* [25]	80.8%/71.4%
STIP + Skeleton	**93.2%/84.6%**

Table 2.13: Comparisons of different methods on MSRDailyActivity3D dataset.

Method	**Accuracy**
NBNN + parts + time [44]	70.0%
Local HON4D [40]	80.0%
DCSF [64]	83.6%
RGGP + Fusion [34]	85.6%
Actionlet [61]	85.8%
DCSF+Joint [64]	88.2%
STIP+Skeleton	80.0%

based action recognition.

From the evaluation, we have shown that most of the results are comparable to the current state-of-the-art approaches. However, under the bag-of-words framework, the extracted features do not contain the spatial distribution of the interest points in depth maps, this is one reason that limits the performance. We have also shown that the noisy depth data and background have a great impact on interest point detection. Moreover, the interest point detection may not perform well on actions without much motion, resulting in lower accuracies.

The evaluation has shown that different STIP features perform quite differently on depth actions. It discovers that the feature with Harris3D and HOG/HOF performs the best on the MSRAction3D dataset, the Cuboids detector with HOG/HOF descriptor performs the best on the MSRDailyActivity3D dataset, while the Harris3D detector combined with HOG3D descriptor is the best on UTKinect-Action dataset. On the CAD-60 dataset, the Hessian detector with ESURF descriptor gives the highest accuracy.

Two interest point refinement schemes have been presented for the STIP features, based on constraining the STIP features using skeleton joint positions and/or the detection in RGB videos. We have shown that the STIP features can be refined to achieve better performance in most cases. We have also proposed a fusion scheme to combine the best STIP features with the skeleton joint features in each database. Significant improvements of the recognition accuracies have been achieved on all four databases. Overall, we have explored the STIP features for 3D action recognition from different aspects.

3

TriViews: A General Framework to Use 3D Depth Data Effectively for Human Action Recognition

3.1 Overview of The Method

Our proposed method contains four steps. First, we use 3D skeleton joints positions to extract the interest region from the original depth map and then project the interest region onto three orthogonal Cartesian planes. Each 4D depth sequence generates three 3D action videos, according to front, side, and top views. After projection, feature extraction is performed on each view separately. Five different features, i.e., spatiotemporal interest points (STIPs), dense trajectory shape descriptor (DT-Shape), dense trajectory motion boundary histograms (DT-MBH), skeleton trajectory shape descriptor (ST-Shape), and skeleton trajectory motion boundary histograms (ST-MBH), are extracted. After feature extraction, we use the Random Forests (RFs) [7] to combine the three views for action recognition. Note that the combination is conducted for each individual feature separately. Finally, we compare the performances of the five features, select the top three best features and fuse them with the probabilistic fusion approach (PFA) [17]. Figure 3.1 illustrates the framework of the proposed approach. The details of each step in our approach will be presented.

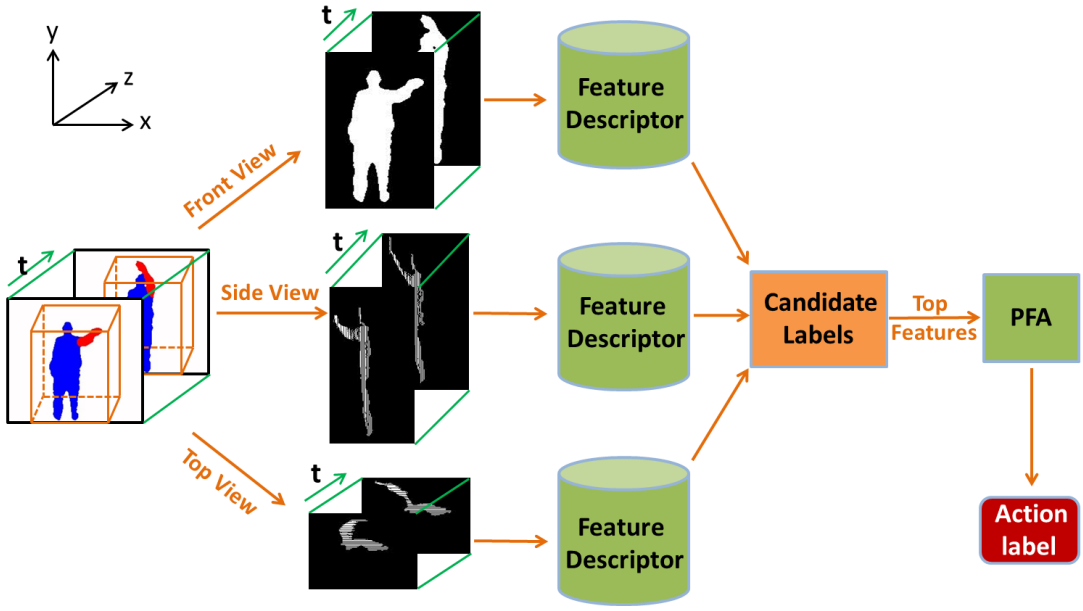

Figure 3.1: The framework of the proposed method. PFA denotes Probabilistic Fusion Approach. Five features, i.e., STIPs, DT-Shape, DT-MBH, ST-Shape, and ST-MBH are used in our experiment.

3.2 TriViews Projection

In order to make use of 3D structure and shape information of depth data, each depth map is projected onto three orthogonal Cartesian planes. We first use the 3D skeleton joints locations to extract a 3D interest region in each action sequence. To be specific, for each depth sequence, we build a 3D bounding box around the human body. In order to construct the bounding box, we first find the boundary positions of the human body in x, y and z directions, respectively and then shift a pixels to both sides in x direction, b pixels in y direction, and c depth units in z direction. These shifts are needed to keep the full motion data for feature extraction during the action executions. Specifically, denote $k = \begin{bmatrix} x_{min} & x_{max} \\ y_{min} & y_{max} \\ z_{min} & z_{max} \end{bmatrix}$ as the boundary of the human body in a video sequence and $K = \begin{bmatrix} X_{min} & X_{max} \\ Y_{min} & Y_{max} \\ Z_{min} & Z_{max} \end{bmatrix}$ the extracted 3D bounding box. We have:

$$K = k + p, \tag{3.1}$$

where $p = \begin{bmatrix} -a & a \\ -b & b \\ -c & c \end{bmatrix}$. After the shifting, if the bounding box goes out of the image plane, we correct it back into the image plane:

$$\begin{cases} \begin{bmatrix} X_{min} \\ Y_{min} \\ Z_{min} \end{bmatrix} = max \begin{bmatrix} X_{min} & 0 \\ Y_{min} & 0 \\ Z_{min} & 0 \end{bmatrix} \\ \begin{bmatrix} X_{max} \\ Y_{max} \\ Z_{max} \end{bmatrix} = min \begin{bmatrix} X_{max} & W \\ Y_{max} & H \\ Z_{max} & D \end{bmatrix} \end{cases} \tag{3.2}$$

where W, H and D are the maximal width, height and depth values of the depth map.

Because (X, Y) are in screen coordinates while Z is in real world coordinate, we first convert Z from real world coordinate to screen coordinate using the linear normalization. Specifically, denote Z_{min} and Z_{max} be the minimum and maximum depth values in a sequence, Z be the depth value, we have:

$$Z^{'} = \frac{255}{Z_{max} - Z_{min}} \times (Z - Z_{min}) \,. \tag{3.3}$$

After the preprocessing, $Z^{'}$ is in the range of [0,255], X, Y, and $Z^{'}$ are all in screen coordinates. We project each frame into three views. Specifically, denote $Q = \left[X, Y, Z^{'}\right]^{'}$ and $q = [x, y, z]^{'}$ the data before and after projection, respectively. Then we have:

$$q = M_i Q, \tag{3.4}$$

where $i \in \{1, 2, 3\}$, $M_i = \begin{bmatrix} f_3 & 0 & 0 \\ 0 & f_2 & 0 \\ 0 & 0 & f_1 \end{bmatrix}$, $f_i = 0$, and $f_j = 1$ $(j \neq i)$, for each projection i.

So each 4D (spatial, depth, and time coordinates) depth sequence can generate three 3D (spatial and time coordinates) action videos. We first extract features on each single view, then combine the three views with the Random Forests (RFs) method [7] on various features.

Note that our TriViews framework is different from [32] and [68], where the depth information in the projection was too simple: only binary images were used for each projection plane. In contrast, we consider the specific depth information and convert the depth values into pixel values in the range of [0,255], so we get real value images

for each single view. Binary maps are limited to a small range of features such as contour shape as in [32] or motion energy images as in [68], while more general features can be extracted from our TriViews framework, resulting in an improved performance significantly.

To demonstrate the effectiveness of our framework, we investigate five different features in our study, which are divided into three categories: STIP, dense trajectory (DT), and skeleton trajectory (ST).

3.2.1 Dense Trajectory

Dense trajectory, proposed by Wang *et al.* [57], was another effective approach for human action recognition in RGB videos. We investigate the dense trajectory feature for 3D actions, under our TriViews framework. All the following computations are on the 3D depth data. Feature points were densely sampled and tracked by median filtering in the dense optical flow field. Specifically, given a point $P_t = (x_t, y_t)$ at frame t, its tracked position at frame $t+1$ is given by:

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * \omega_t)\,|_{(x_t, y_t)}, \tag{3.5}$$

where M is the 3×3 median filtering kernel, $\omega_t = (u_t, v_t)$ denotes the optical flow field computed by [14]. To avoid the drifting problem, feature points were only tracked for 15 frames and new points were sampled. In order to increase the precision of the trajectories, both static trajectories and trajectories with large displacements were pruned in a post-processing stage.

To characterize the actions, two categories of feature were used: (1) Trajectory shape feature; (2) Local motion and appearance descriptors, i.e., histograms of gradients (HOG), histograms of optical flow (HOF), and motion boundary histograms (MBH).

The shape of a trajectory was described by a sequence of displacement vectors: $(\triangle P_t, ..., \triangle P_{t+L-1})$, where $\triangle P_t = (P_{t+1} - P_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$, L is the trajectory length. Specifically, we concatenate L displacement vectors to get a $2 \times L$ feature vector.

To make the trajectory shape descriptor invariant to scale changes, the concatenated feature vector is normalized by the overall magnitude of motion displacements:

$$\text{DT-Shape} = \frac{(\triangle P_t, ..., \triangle P_{t+L-1})}{\sum_{j=t}^{t+L-1} \| \triangle P_j \|}, \tag{3.6}$$

where $L = 15$ is the trajectory length.

HOG, HOF, and MBH descriptors are computed in trajectory-aligned 3D video volumes of size $32 \times 32 \times 15$. HOG encodes local appearance information, while HOF and MBH capture local motion pattern. All the histogram features are normalized with the L_2 norm. To further embed the structure information, the 3D volume is subdivided into a spatiotemporal grid of size $2 \times 2 \times 3$. Descriptors (e.g., HOG,HOF or MBH) are computed in each cell of the spatiotemporal grid, and the final descriptor is a concatenation of descriptor from each cell. HOG is quantized into 8 bins and HOF into 9 bins. MBH is a concatenation of two components, i.e., MBHx and MBHy. MBHx is the derivative for the horizontal component of the optical flow, while MBHy is the derivative for the vertical component. MBH has been shown to outperform both HOG and HOF descriptors in almost all cases [57], so we investigate MBH descriptor in our study of 3D action videos.

3.2.2 Skeleton Trajectory

Although the dense trajectory can have a good coverage of foreground motion as well as the surrounding context, it is time-consuming to track the densely sampled feature points. So we propose a 3D sparse trajectory called skeleton trajectory (ST) for depth-based action recognition, which is faster than the dense trajectory.

RGB-D sensors, such as the Kinect, provide twenty 3D skeleton joint positions in real time [46]. The skeleton joints are a natural representation of sparse trajectories. Specifically, denote $Q_i(t) = (x_i(t), y_i(t), z_i(t))$ the ith skeleton joint at frame t and the number of skeleton joints in each frame as N, ST can be denoted as:

$$(Q_i(t), Q_i(t+1), ..., Q_i(t+L-1)), \tag{3.7}$$

where L is the trajectory length, $i = 1, 2, ..., N$.

Note that the location (x_i, y_i, z_i) of a joint Q_i might be of inconsistent coordinates. For example, (x_i, y_i) are in screen coordinates, while z_i is in real world coordinate. We convert z_i from real world coordinate to screen coordinate. Empirically, each depth value is scaled to the range of [0,255] with a linear normalization scheme.

Also note that the proposed skeleton trajectory is different from the spatiotemporal motion trajectories proposed by Devanne *et al.* [11]. They concatenated the coordinates of all 20 skeleton joints to form a 60 dimensional vector, and the evolution of such a

vector over time was used as a trajectory, while we treat 20 skeleton joints separately and each skeleton joint counterparts one trajectory.

We compute two features for skeleton trajectory: skeleton trajectory shape descriptor (ST-Shape) and motion boundary histograms (ST-MBH). ST-Shape characterizes trajectory shape and motion information, while ST-MBH encodes local appearance and motion patterns.

The ST-Shape feature is computed as:

$$\text{ST-Shape}_i = \frac{(\triangle Q_i(t), ..., \triangle Q_i(t+L-1))}{\sum_{j=t}^{t+L-1} \|\triangle Q_i(j)\|}, \tag{3.8}$$

where L is the length of skeleton trajectory, $i = 1, 2, ..., 20$.

To encode the local motion information, we compute the MBH feature aligned with the skeleton trajectory. Specifically, denote I^x and I^y as images containing the horizontal and vertical components of optical flow computed from 3D depth data, respectively, we take their local gradients separately, find the corresponding bins, compute the weighted votes, and build histograms using the weighted votes into local orientation histograms. Finally, we combine the two components to get the ST-MBH feature.

Note that skeleton trajectory features are unique for depth data. Both the ST-Shape and ST-MBH are new features to characterize the 3D depth data using the skeleton trajectories. These features cannot be computed as the dense trajectories in RGB videos as in [57].

3.3 Random Forests

After obtaining features from each single view, we propose to combine three views with the Random Forests (RFs) method [7].

Random Forests are a collection of many decision trees. For a test sample x, each tree gives a classification decision, and the final classification result is the class label which gets the most votes from all the trees. The forest grows as follows:

Denote the feature vector as $v \in R^N$, where N is the feature dimension for each sample. At each node n features are selected out of N at random to split the node. The best split is determined by the information gain using the n selected features:

$$Gain = \sum_{i=1}^{2} \left(\frac{|I_i|}{I} \sum_{j=1}^{C} p_{i,j} log_2(p_{i,j}) \right), \tag{3.9}$$

where I_i are the two splits I_{left} and I_{right}, $|I_i|$ is the size of set I_i, C is the total class types, $p_{i,j}$ is the fraction of samples in I_i belonging to class j. Once the best split is found, a binary split is performed on that node. At the next node, another n features are chosen randomly and the same procedure is performed. Each tree grows until it reaches the maximum tree depth max_{depth}, or the tree node receives the predefined number of minimum samples min_{node}. In the leaf node, the probability distribution for each class is computed.

3.4 Probabilistic Fusion Approach

After evaluating five different features under the TriViews framework, we propose to fuse the top three best features on each database with the probabilistic fusion approach (PFA), proposed in [17]. It was originally used to fuse regressors and classifiers for human age estimation. We adapted the approach to combine the outputs of multiple Random Forests (RFs) in our study of 3D action recognition.

Let $\mathbf{v} \in \Re^n$ be a feature vector extracted from an input pattern, and let $\{c_1, c_2, ..., c_n\}$ be the class labels of n classes. For each Random Forests $D_i, i = 1, 2, ..., L$, the output of D_i given the input pattern v is: $D_i(v) = [N_{c_1}(v), N_{c_2}(v), ..., N_{c_n}(v)]$, where $N_{c_j}(v), j = 1, 2, ..., n$ is the number of trees that classify v into class c_j. We can create a probability measure for the RFs output as:

$$P_i\left(c_j|v\right) = \frac{N_{c_j}\left(v\right)}{\sum_{j=1}^{n} N_{c_j}\left(v\right)}, \tag{3.10}$$

Let $p_i\left(c_j|v\right)$ denote the probability measure that classifier D_i classify v into class c_j, then the probabilistic fusion approach is given by

$$p\left(c_j|v\right) = A \cdot \sum_{i=1}^{L} \omega_i p_i\left(c_j|v\right), \tag{3.11}$$

where ω_i is the weight for classifier D_i and $\sum_{i=1}^{L} \omega_i = 1$. A is a constant to maintain a probabilistic measure.

3.5 Experiments

We conduct experiments on three challenging depth-based action recognition databases, and compare our results with the state-of-the-art methods on each database.

For each action sequence, five features, i.e., the STIP, DT-Shape, DT-MBH, ST-Shape, and ST-MBH will be employed to encode the motion and local appearance information. We evaluate the features in a bag-of-features scheme. Vocabularies are constructed with the K-means clustering. For each single view, i.e., front, side, or top view, the SVMs is used as the classifier. When combining three views, RFs is used, which can also do feature selection based on the randomized mechanism.

In the following, we introduce our experimental settings, and then present the experimental results. Finally we provide analysis and discussions, and compare to the state-of-the-art methods.

3.5.1 Experimental Settings

When extracting the 3D bounding box of the human body, we shift 50 pixels in x direction, 15 pixels in y direction and 120 depth units in z direction. For STIP features, we select the best detector and descriptor for each database. Specifically, Harris3D detector and HOG/HOF descriptor are used for MSRAction3D dataset. Harris3D detector combined with HOG3D descriptor are used for UTKinect-Action3D dataset. On the MSRDailyActivity3D dataset, Hessian detector and HOG3D descriptor are employed to extract local features. These selections are the best STIP features based on a comparison among various STIP features [72]. The K-means clustering method is applied to quantize the STIP features into histograms. Because of different application scenarios, the number of visual words V is selected from [500, 2000], with a step size of 100. To limit the complexity, we cluster a subset of 100,000 randomly selected training features. To increase precision, we initialize K-means 5 times and keep the result with the lowest error. In order to get the dense trajectory feature, we use the settings as [57] although the 3D data are different from the color videos in [57]. The trajectory length L is 15 frames (6 frames for UTKinect-Action dataset because this dataset has much shorter video clips). The size of space-time volume aligned with a trajectory is set to $32 \times 32 \times 15$. To embed structure information, each volume is subdivided into a spatiotemporal grid of size $2 \times 2 \times 3$. For skeleton trajectory, we adopt the same parameter settings as the dense trajectory. For the classifiers used in the experiment, SVMs with χ^2 kernel are used. For the random forests, the number of trees can be selected from $[1, 500]$, and the number of features used in each split can be selected from $[3, 60]$.

3.5.2 Experimental Results

We present the experimental results of individual features, and then give the fusion results using the top three best features.

3.5.2.1 Spatiotemporal Features

We first investigate spatiotemporal features under our new framework. The bag-of-words approach is used for histogram construction and SVMs is used as the classifier. When combining three views, the Random Forests (RFs) method is used, but because the SVM classifier is used for each single view, in order to have a fair comparison, we also conduct the experiment of combining the three views using the SVM classifier.

The experimental results using STIP features on three databases are shown in Table 3.1. We can see that, the front view can get better results than the other two views on the MSRAction3D (Accuracy: 90.0%) and UTKinect-Action database (Accuracy: 84.8%), while on the MSRDailyActivity3D database, the side view gets a better result (Accuracy: 73.8%). One reason why the side view has a better performance than the front view on the MSRDailyActivity3D database might be that the actions in this database look more different from the side view. After combining three views with RFs, the accuracies can be improved to 94.9% on MSRAction3D dataset, 92.9% on UTKinect-Action dataset, and 83.8% on MSRDailyActivity3D database. We get higher accuracies after combining three views. It validates the usefulness of our proposed framework that combining three views can improve the overall performance significantly.

To further demonstrate the effectiveness of the TriViews framework, we compare our method with [71]. In [71], the 3D depth data were transformed into gray level depth videos, extracted spatiotemporal feature on depth video and finally did classification in a bag-of-words scheme. We use the approach in [71] with our experimental settings and present the results in the first two rows of Table 3.1. To have a fair comparison, we keep the same parameter settings and the same training and test data sets. From the table, it can be seen that our method outperforms [71] on all three databases. When the classifier is the SVM, our method performs 1.6%, 5.1%, and 10.0% higher than [71] on the MSRAction3D, UTKinect-Action and MSRDailyActivity3D databases, respectively. When the classifier is RFs, the improvements are even higher (2.1% on

MSRAction3D dataset, 9.1% on UTKinect-Action dataset, and 15.0% on MSRDaily-Activity3D dataset). The results validate the effectiveness of the proposed method, i.e., the recognition accuracies can be improved for the spatiotemporal features using the TriViews framework.

Table 3.1: Performances of STIP with and without projecting the 3D depth data into three views on three databases: MSRAction3D, UTKinect-Action and MSRDailyActivity3D. Depth denotes gray level depth videos. TriViews denotes combining three views.

Method		**Accuracy**		
		MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
Without Projection	Depth + SVM	89.7%	83.8%	65.0%
	Depth + RFs	92.8%	83.8%	68.8%
With Projection	Front + SVM	90.0%	84.8%	65.0%
	Side + SVM	88.2%	78.8%	73.8%
	Top + SVM	86.9%	77.8%	61.9%
	TriViews + SVM	91.3%	88.9%	75.0%
	TriViews + RFs	**94.9%**	**92.9%**	**83.8%**

3.5.2.2 Comparison to HON4D Feature

Projecting depth data into three views has shown superiority over the standard use of the 3D depth data (i.e., without projection). To further demonstrate the effectiveness of the TriViews framework for depth-based action recognition, we compare our results with [40], which is a representative work that characterizes the depth data as a surface in 4D space. Histogram of the 4D surface normals (HON4D) over all voxels in the depth sequence is used for action recognition. To the best of our knowledge, HON4D is currently the most effective feature to characterize depth data in 4D. In order to have a fair comparison, we run the code provided by [40] on the three datasets, using the same experimental conditions. The results are shown in Table 3.2.

Table 3.2: Comparison of our proposed method with HON4D feature on the three databases.

Method	**Accuracy**		
	MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
HON4D [40]	92.0%	77.8%	75.6%
TriViews+STIPs+RFs	**94.9%**	**92.9%**	**83.8%**

From Table 3.2, it can be seen that our approach outperforms the HON4D feature on all three databases. Compared with the HON4D feature, our approach performs 2.9% higher on MSRAction3D, 15.1% higher on UTKinect-Action, and 8.2% higher on MSRDailyActivity3D, respectively.

After evaluating our proposed TriViews framework with the STIP features, we explore the framework further with more features, i.e., DT-Shape, DT-Motion, ST-Shape, and ST-MBH, and show the performance using the same data (training and test sets).

3.5.2.3 Dense Trajectory

Table 4.3 shows the experimental results on three databases using dense trajectory shape (DT-Shape) and dense trajectory motion boundary histograms (DT-MBH) features. We can see that for both DT-Shape and DT-MBH features, combining three views gets higher accuracies than any single view. It can also be observed that MBH consistently outperforms trajectory shape descriptor. On the MSRAction3D database, trajectory shape descriptor can get an accuracy of 84.8%, while MBH descriptor gets an accuracy of 96.4%. On the other two databases, trajectory shape descriptor gets an accuracy of 84.8% and 75.6%, respectively, while MBH gets better results (90.9% on UTKinect-Action and 87.5% on MSRDailyActivity3D, respectively).

After evaluating dense trajectory, we conduct experiments using skeleton trajectory, and then compare these two kinds of trajectory features.

Table 3.3: Performances of dense trajectory shape (DT-Shape) descriptor and motion boundary histograms (DT-MBH) feature on each single view and three views combination.

Methd		**Accuracy**		
		MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
Front	DT-Shape	79.0%	78.8%	65.0%
	DT-MBH	89.3%	81.8%	66.3%
Side	DT-Shape	63.9%	78.8%	60.6%
	DT-MBH	90.9%	79.8%	67.5%
Top	DT-Shape	77.0%	64.6%	58.1%
	DT-MBH	92.4%	70.7%	59.4%
TriViews	DT-Shape	84.8%	84.8%	75.6%
	DT-MBH	**96.4%**	**90.9%**	**87.5%**

3.5.2.4 Skeleton Trajectory

The results of skeleton trajectory shape (ST-Shape) and skeleton trajectory motion boundary histograms (ST-MBH) features on the three databases are shown in Table 3.4. We can see that combining three views can improve recognition rate on all three databases. MBH has a better performance than the trajectory shape descriptor on the MSRAction3D database, while on the other two databases skeleton trajectory shape descriptor outperforms MBH feature. One reason why the MBH performs better than skeleton trajectory shape on the MSRAction3D database might be that this database has a clean background, thus local appearance descriptor such as MBH can characterize the actions more reliably. Skeleton trajectory shape descriptor gets an accuracy of 89.9% on the UTKinect-Action and 77.5% on MSRDailyActivity3D, respectively.

Table 3.4: Performances of skeleton trajectory shape (ST-Shape) descriptor and motion boundary histograms (ST-MBH) feature on each single view and three views combination.

Methd		**Accuracy**		
		MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
Front	ST-Shape	78.0%	76.8%	69.4%
	ST-MBH	85.4%	66.7%	55.6%
Side	ST-Shape	71.4%	84.8%	70.0%
	ST-MBH	76.6%	70.7%	58.1%
Top	ST-Shape	68.0%	76.8%	70.6%
	ST-MBH	79.6%	65.7%	58.1%
TriViews	ST-Shape	86.8%	**89.9%**	**77.5%**
	ST-MBH	**94.2%**	88.9%	71.3%

3.5.2.5 Comparison of Dense Trajectory with Skeleton Trajectory

Dense trajectory collects the video motion and appearance information densely, while skeleton trajectory is a sparse representation of motion and local appearance information. We compare these two trajectories in Table 3.5. One can see that for the trajectory shape descriptor, skeleton trajectory outperforms dense trajectory on all three databases, while for the MBH descriptor, dense trajectory consistently outperforms skeleton trajectory. ST-MBH gets an accuracy of 94.2%, 88.9%, and 71.3% respectively on the three databases, while DT-MBH gets an accuracy of 96.4%, 90.9%, and

87.5%, respectively. Among the four features, DT-MBH can get the highest accuracy on all three databases.

Table 3.5: Comparison of dense trajectory (DT) with skeleton trajectory (ST) on three databases.

Method		**Accuracy**		
		MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
Shape	DT	84.8%	84.8%	75.6%
	ST	86.8%	89.9%	77.5%
MBH	DT	96.4%	90.9%	87.5%
	ST	94.2%	88.9%	71.3%

3.5.2.6 Comparison of Different Skeleton Based Features

From previous results we can see that skeleton trajectory shape descriptor not only consistently outperforms dense trajectory shape descriptor, but also exhibits better performance than ST-MBH feature on UTKinect-Action and MSRDailyActivity3D databases. ST-Shape feature uses only the 3D skeleton location information, we compare this feature with other published work using skeleton locations for depth action recognition in Table 3.6. Note that on UTKinect-Action and MSRDailyActivity3D datasets, we implement [67]'s work and get the results with our experimental settings. From Table 3.6 it can be seen that on the MSRAction3D and MSRDailyActivity3D databases, the accuracy of ST-Shape descriptor is 86.8% and 77.5%, respectively, higher than all the other published results. On the UTKinect-Action database, ST-Shape gets an accuracy of 89.9%, higher than 87.9% by EigenJoints [67]. HOJ3D [65] gets an accuracy of 90.92%, but they employed a leave-one-out setting, where more training samples and less test sample were used in their experiment. The cross-subjects setting is applied in our experiment.

3.5.2.7 Fusing Top Three Features

From above results, we found that among the five individual features, STIPs and DT-MBH consistently outperform the other three features. On MSRAction3D dataset, the third best feature is ST-MBH, while on the other two datasets ST-Shape outperforms both DT-Shape and ST-MBH features. A fusion scheme is proposed to combine the top three best features based on the probabilistic fusion approach (PFA) method [17].

Table 3.6: Comparison of ST-Shape feature with other published results using skeleton-based features for 3D action recognition. *Note that, we use half subjects for training and the other half for testing. In [65], a leave-one-out setting was applied.

Method	Accuracy		
	MSRAction3D	**UTKinect**	**MSRDailyActivity3D**
ST-Shape	**86.8%**	**89.9%**	**77.5%**
EigenJoints[67]	82.3%	87.9%	73.8%
HOJ3D[65]	78.97%	90.92%*	-
Actionlet[61]	-	-	68.0%
Key Pose Learning[31]	80.3%	-	-

The experimental results on MSRAction3D dataset are shown in Table 3.7. The accuracy is only 94.2% using ST-MBH feature, 94.9% using only STIP and 96.4% using DT-MBH. But after fusion with PFA, recognition rate can be improved to 98.2%, which is higher than each of the individual features.

Table 3.7: The recognition accuracies of top three individual features and fusion by the probabilistic fusion approach (PFA). Note that on MSRAction3D dataset, the third best feature is ST-MBH while on the other two datasets, the third best feature is ST-Shape.

Method		Accuracy		
		MSRAction3D	**UTKinect-Action**	**MSRDailyActivity**
Single Feature	STIPs	94.9%	92.9%	83.8%
	DT-MBH	96.4%	90.9%	87.5%
	ST-MBH/ST-Shape	94.2%	89.9%	77.5%
	Average	95.2%	91.2%	82.9%
Fusion	PFA	**98.2%**	**98.0%**	**88.8%**

The results on the UTKinect-Action dataset are shown in the second column of Table 3.7. One can see that after fusing the three individual features, the accuracy achieves 98.0%, which is significantly higher than any single feature. For example, the accuracy of the best individual feature STIP is 92.9%.

The results on the MSRDailyActivity3D dataset are shown in the third column of Table 3.7. The accuracy is only 77.5% using ST-Shape feature, 83.8% using only STIP and 87.5% using DT-MBH, but after fusion, the accuracy achieves 88.8%. The results validate the effectiveness of fusing multiple features to improve the performance for 3D action recognition.

Table 3.8: Comparison of the recognition accuracies between our approach and all existing methods on MSRAction3D dataset.

Method	**Accuracy**
High Dimensional Convolutional Network [60]	72.5%
Action Graph [32]	74.7%
HOJ3D [65]	79.0%
Key Pose Learning [31]	80.3%
Sparse Representation [4]	80.8%
OESGP [47]	80.9%
EigenJoints [67]	82.3%
STOP [54]	84.8%
ROP [60]	86.2%
Actionlet [61]	88.2%
HON4D [40]	88.9%
DSTIP+DCSF [64]	89.3%
Part-set [56]	90.2%
Depth Motion Maps [68]	91.6%
Space-time pose representation [11]	92.8%
Evolutionary Joint Selection [10]	93.2%
DS-SRC [52]	93.6%
STIPs+Joint+RFs [71]	94.3%
JAS (Cosine)+MaxMin+HOG^2 [39]	94.8%
TriViews + ST-MBH	94.2%
TriViews + STIPs	**94.9%**
TriViews + DT-MBH	**96.4%**
TriViews + PFA	**98.2%**

Table 3.9: Comparison of the recognition accuracies between our approach and all existing methods on the UTKinect-Action dataset. Note that, we used a less number of training examples, while the leave-one-out setting was used in [65].

Method	**Accuracy**
Posture Word [64]	79.57%
DSTIP+DCSF [64]	85.8%
HOJ3D [65]	90.9%
DS-SRC [52]	91.0%
Space-time pose representation [11]	91.5%
STIPs+Joint+RFs [71]	91.9%
TriViews + ST-Shape	89.9%
TriViews + DT-MBH	90.9%
TriViews + STIPs	**92.9%**
TriViews + PFA	**98.0%**

Table 3.10: Performance comparison between our approach and state-of-the-art methods on MSRDailyActivity3D dataset. Note that, all the actions are used in our experiment, while in [64] four actions (with less motion) were removed from the dataset in their experiment.

Method	**Accuracy**
NBNN+parts+time [44]	70.0%
Local HON4D [40]	80.0%
DCSF [64]	83.6%
RGGP+Fusion [34]	85.6%
Actionlet [61]	85.8%
DCSF+Joint [64]	88.2%
TriViews + ST-Shape	77.5%
TriViews + STIPs	83.8%
TriViews + DT-MBH	87.5%
TriViews + PFA	**88.8%**

3.5.2.8 Computation Complexity

We first evaluate the run time of different features used in our experiments. We compute the three kinds of features for all the training video samples from the MSRAction3D dataset. There are about 14,000 frames in the training data. We use the STIP toolbox from [30] with the default settings. For dense trajectory features, we use the toolbox from [57]. The skeleton trajectory features were implemented in C++. Experiments were conducted on a Dell desktop with a 3.4 GHz Intel Core i7-2600 CPU and 12GB RAM.

From Figure 3.2, we can see that STIP gets the run time of 2.19fps, while dense trajectory features has the speed of 1.64fps. Skeleton trajectory can get 2.08fps, which is faster than the dense trajectory features, a little slower than STIP feature.

Besides feature extraction, we also list the run time for other steps in our experiments, i.e., three views projection, action recognition using the features extracted from the three views, and fusing top three features with PFA for final action classification. The three views projection is implemented in Matlab and the run time for this step is 1.21fps on average. When combining three views, we concatenate the features from three views and then use random forests for classification. And the run time for this step is 3.34vps (videos per second). When fusing top three features with PFA, it is almost real time. For example, it takes about 0.08 seconds for every 100 test samples.

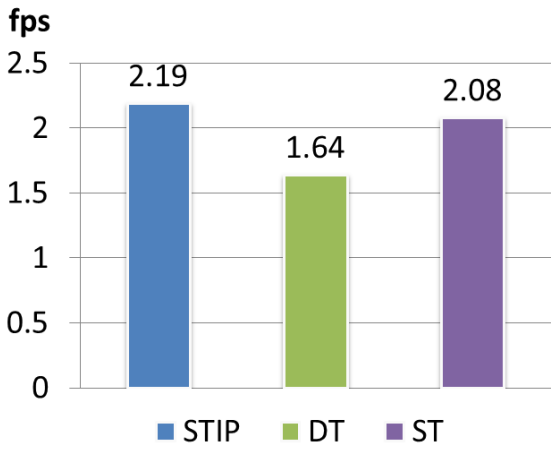

Figure 3.2: Run time for different features. We use Harris3D+HOG/HOF for STIP. ST denotes skeleton trajectory. DT denotes dense trajectory. Note that for both ST and DT, we compute two features: MBH and shape descriptors.

3.5.3 Comparison with the State-of-the-art Methods

We further compare our approach with the state-of-the-art methods for depth-based action recognition on the three challenging datasets. Note that we adopt cross-subjects scheme in all our experiments, where half of the subjects are used for training and the remaining half for testing. We list all the published results on the three databases, to the best of our knowledge. Table 3.8 shows the reported results in the literature on the MSRAction3D dataset. We can see that under our TriViews framework, STIP feature achieves an accuracy of 94.9%, and DT-MBH gets an accuracy of 96.4%, both are higher than all of the previously reported results. After fusion with PFA, the accuracy can be improved to 98.2%, which is 3.4% higher than the best result 94.8% [39] in the literature. On the UTKinect-Action dataset, the results are shown in Table 3.9. STIP feature gets an accuracy of 92.9% and PFA achieves 98.0%, both are higher than the state-of-the-art methods on this dataset. Finally, we compare our results with the state-of-the-art on the MSRDailyActivity3D dataset. From Table 3.10 one can see that, after fusion, the accuracy achieves 88.8% , which outperforms the DCSF+Joint approach in [64]. Also note that DT-MBH can get an accuracy of 87.5% under TriViews framework, which is close to the 88.2% reported in [64], however, four still actions were eliminated in their experiments, while we use all 16 actions.

3.6 Conclusions

We have proposed a general framework for 3D depth-based action recognition called TriViews. It can utilize the rich 3D information effectively for action recognition. Under this framework, we have investigated five different features: three features are adapted from representative approaches in RGB videos, and the other two are proposed uniquely for depth-based action recognition. The top three best features are combined by a probabilistic fusion approach (PFA). The experimental results demonstrate that the TriViews framework is very effective to improve the 3D action recognition performance, outperforming the state-of-the-art results on each of the three challenging databases.

4

Mice Behavior Recognition based on Integration of Sparse and Dense Trajectory Features

4.1 Overview of The Method

Our proposed approach contains four steps. First, the sparse trajectory features (STF) are computed from tracked mice positions. Then dense trajectory features (DTF), including dense trajectory shape (DT-Shape) and dense trajectory motion boundary histograms (DT-MBH) features are extracted from the mice videos. After feature extraction, we do mice action recognition for each feature separately, thus get three sets of action candidates. Finally, we fuse action candidates obtained from different features with several fusion methods. Figure 4.1 illustrates the framework of the proposed approach. The details of each step are presented in the following.

4.2 Sparse Trajectory Features

Inspired by Burgos-Artizzu et al.'s work on trajectory features (TF) [9] and Giancardo et al.'s work of spatiotemporal features [16], we consider three kinds of features from mice positions: (1) Zero-order position information: position, distance, and direction; (2) First-order position information: velocity; (3) Second-order position information: acceleration. Position and distance information help discriminate between solitary and social behaviors (e.g., clean/approach). Direction feature helps discriminate behaviors

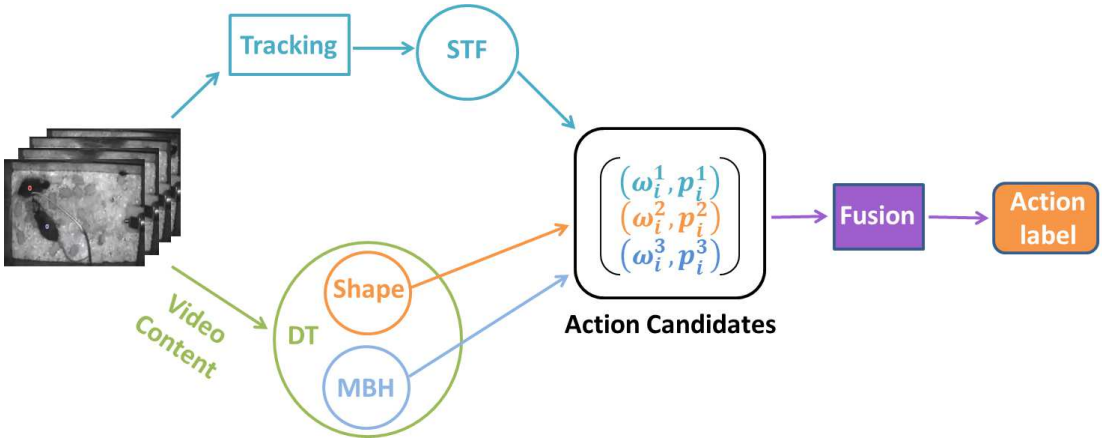

Figure 4.1: The framework of the proposed method. STF means sparse trajectory features, DT means dense trajectory, MBH means motion boundary histograms, $\left(\omega_i^j, p_i^j\right)$ denotes the sample assigned to class label ω_i by classifier j with a confidence p_i.

with different interactions (e.g., nose to nose/nose to genital). Velocity and acceleration are important for distinguishing stationary behaviors from those with large motions. TF [9] computes the above trajectory feature at a local region and sums them together, but we use the trajectory features in a different way. We concatenate the three kinds of feature vectors and then apply a gaussian normalization to normalize the feature values to the range of $[0, 1]$. Our sparse trajectory features are also different from spatiotemporal features [16], which computes some distance and shape information between two mice, while our sparse trajectory features (STF) include some velocity and acceleration features.

Specifically, for two interacting mice, denote $(x_{m_i}(t), y_{m_i}(t))$ as the position for mouse $m_i \in [1, 2]$ at frame t. Each feature can be denoted as:

(1a) Position: $x_{m_i}(t), y_{m_i}(t)$

(1b) Distance between two mice:

$$Dist(t) = \sqrt{(x_{m_1}(t) - x_{m_2}(t))^2 + (y_{m_1}(t) - y_{m_2}(t))^2} \tag{4.1}$$

Inspired by the "relative position" feature in [9], when one mouse is tracked with 3 points: head, body, and genital, we add 4 more distance features: "head2head", "head2body", "head2genital", and "genital2genital".

(1c) Distance change between two consecutive frames:

$$DistChange(t) = Dist(t) - Dist(t-1) \tag{4.2}$$

(1d) Direction for each mouse:

$$Dir_{m_i} = atan^{-1}\left(\frac{y_{m_i}(t) - y_{m_i}(t-1)}{x_{m_i}(t) - x_{m_i}(t-1)}\right) \tag{4.3}$$

(1e) Direction change between two consecutive frames:

$$DirChange_{m_i}(t) = Dir_{m_i}(t) - Dir_{m_i}(t-1) \tag{4.4}$$

(1f) Direction difference between two mice:

$$DirDiff(t) = \mid Dir_{m_1}(t) - Dir_{m_2}(t) \mid \tag{4.5}$$

(2) Velocity for each mouse:

$$\begin{bmatrix} V_{x_{m_i}}(t) \\ V_{y_{m_i}}(t) \end{bmatrix} = \frac{1}{\triangle t} \begin{bmatrix} \triangle x_i(t-1) & \triangle x_i(t) & \triangle x_i(t+1) \\ \triangle y_i(t-1) & \triangle y_i(t) & \triangle y_i(t+1) \end{bmatrix} \cdot \begin{bmatrix} 0.25 \\ 0.50 \\ 0.25 \end{bmatrix} \tag{4.6}$$

,where $\triangle x_i(t_0) = x_{m_i}(t_0) - x_{m_i}(t_0 - 1)$, and $\triangle y_i(t_0) = y_{m_i}(t_0) - y_{m_i}(t_0 - 1)$.

(3) Acceleration for each mouse:

$$\begin{cases} A_{x_{m_i}}(t) = \frac{V_{x_{m_i}}(t+1) - V_{x_{m_i}}(t-1)}{2\triangle t} \\ A_{y_{m_i}}(t) = \frac{V_{y_{m_i}}(t+1) - V_{y_{m_i}}(t-1)}{2\triangle t} \end{cases} \tag{4.7}$$

The total dimension for the sparse trajectory features is 19 without the "relative position" features. After adding the "relative position" distance features, the final feature dimension is 23.

4.3 Dense Trajectory Features

Dense trajectory features were proposed by Wang *et al.* [57] for human behavior recognition. Feature points were densely sampled with a step size of 5 pixels. In order to guarantee a good coverage of the video content, 8 spatial scales were employed. Sampling was carried out on each spatial scale, separately. Tracking was performed by median filtering in the optical flow field. Specifically, given a point $P_t = (x_t, y_t)$ at frame t, the corresponding position at frame $t+1$ is given by:

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * \omega_t)\mid_{(x_t, y_t)}, \tag{4.8}$$

where M is the 3×3 median filtering kernel, $\omega_t = (u_t, v_t)$ denotes the optical flow field computed by the method in [14], which was implemented in the OpenCV library. Tracked points of subsequent frames were concatenated to form trajectories: $(P_t, P_{t+1}, P_{t+2}, ..., P_{t+L-1})$. In order to avoid drifting, only the tracked feature points for 15 frames are used and new points will be sampled to replace them. To increase the precision of the trajectories, both static trajectories and trajectories with sudden large displacements were pruned in a post-processing stage.

To encode the video sequences, two kinds of features will be considered: (1) Point locations to characterize trajectory shape and motion information; (2) Feature descriptors to encode local motion and local appearance information.

To be specific, the shape of a trajectory is described by a sequence of displacement vectors: $(\triangle P_t, ..., \triangle P_{t+L-1})$, where $\triangle P_t = (P_{t+1} - P_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$, and L is the trajectory length. To make the trajectory shape descriptor invariant to scale changes, the concatenated feature vector is normalized by the overall magnitude of motion displacements:

$$DT\text{-}Shape = \frac{(\triangle P_t, ..., \triangle P_{t+L-1})}{\sum_{j=t}^{t+L-1} \| \triangle P_j \|}, \tag{4.9}$$

where $L = 15$ is the length of trajectory.

The local motion and appearance information is described by trajectory-aligned features such as histograms of oriented gradients (HOG), histograms of optical flow (HOF), and motion boundary histograms (MBH). HOG encodes local appearance information, while HOF and MBH capture local motion pattern. All the histogram features are normalized with the L_2 norm. A 3D video volume of size $32 \times 32 \times 15$ is aligned with each trajectory and features are computed in the 3D volume. To further embed the structure information, the 3D volume is subdivided into a spatiotemporal grid of size $2 \times 2 \times 3$. Descriptors (e.g., HOG, HOF or MBH) are computed in each cell of the spatiotemporal grid, and the final descriptor is a concatenation of descriptor from each cell. HOG is quantized into 8 bins and HOF into 9 bins. MBH has two components, i.e., MBHx and MBHy. MBHx is the derivative for the horizontal component of the optical flow, while MBHy is the derivative for the vertical component. Each component (i.e., MBHx and MBHy) is quantized into 8 bins. So the dimension is $2 \times 2 \times 3 \times 8 = 96$ for both MBHx and MBHy. MBH is a concatenation of MBHx and MBHy. MBH has been shown to

outperform both HOG and HOF descriptors for human action recognition [57]. Here we investigate the DT-MBH feature for mice behavior analysis.

4.4 Post-Processing with Temporal Coherence Features (TCF)

For either sparse trajectory features or dense trajectory features, we use Random Forests [7] to do frame-based action recognition, so each frame t will have a confidence vector $[h_1(t), h_2(t), ..., h_K(t)]$, where K is the total behavior types. Because temporally close frames are very likely to have the same behavior type, inspired by the auto-context work in [9], we consider a post-processing approach and compute three kinds of features from the confidence scores of frames preceding and following the current frame t: pairwise confidence difference between actions, first order derivative of confidence score, and some statistical features including mean, maximum, minimum, and variance values.

Note that in order to search the starting point of a new behavior and to detect the ending point of an existing behavior, both the statistical features and the first order derivative features are computed in three types of windows:

(a) Center_frame window: $\triangle W_1 = \left[t - \frac{sz}{2}, t + \frac{sz}{2}\right]$;

(b) Pre_frame window: $\triangle W_2 = \left[t - \frac{sz}{2}, t\right]$;

(c) Past_frame window: $\triangle W_3 = \left[t, t + \frac{sz}{2}\right]$;

The temporal coherence features are added to the original STF, DT-Shape or DT-MBH feature sets for a new iteration of computation. This process is repeated 2 times in our experiments.

4.5 Gaussian Normalization

When adding temporal coherence measure to either STF, DT-Shape, or DT-MBH features, we use the gaussian normalization to map different features into a comparable range. Suppose there are M samples in the database, after adding the temporal coherence measure we can get an $M \times N$ feature matrix $F = f_{ij}$, where f_{ij} is the jth feature component in feature vector $f_{i,\cdot}$, each feature vector is of N dimensions. Our goal is to normalize the entries in each column $f_{\cdot,j}$ to the same range so as to ensure

that each individual feature component is within the same range in determining the similarity between two vectors. We compute the mean μ_j and standard deviation σ_j of each feature, and then normalize the original features into a normal distribution $N \sim (0, 1)$ as follows:

$$f'_{ij} = \frac{f_{ij} - \mu_j}{3\sigma_j} \tag{4.10}$$

then, the probability of a feature component value in the range of [-1, 1] is approximately 99%. An additional shift will guarantee that 99 percent of feature values are within [0,1]:

$$\tilde{f_{ij}} = \frac{f'_{ij} + 1}{2} \tag{4.11}$$

After this shift, we can consider that all of the feature component values are within the range of [0,1]. Therefore, this normalization process ensures the same range of the feature components when different types of features are combined.

4.6 Fusion Methods

Data fusion has received considerable attention in recent years. Decision level fusion is one of the most popular fusion types [23, 26]. Given a set of classifiers $\{D_1, D_2, ..., D_L\}$, decision level fusion aims at a higher accuracy than any single classifier. We studied five representative classifier fusion strategies [2]: minimum, maximum, sum(average), median, and majority voting, and chose the majority voting and median based fusion methods because of their simplicity and good performance in our experiments.

4.6.1 Majority Voting

Majority voting [23] is one of the most common approaches for decision level fusion. The idea is that in combining the decisions of multiple classifiers, the sample is assigned the label that the majority classifiers agree with. Specifically, for an input sample x, each classifier D_i , $i = 1..N$, outputs a predicted class label ω_i^c. The final class label is the one with the most occurrence in the decision vector $\Omega = \begin{bmatrix} \omega_1^c & \cdots & \omega_i^c & \cdots & \omega_N^c \end{bmatrix}$. If two or more labels have the most equal occurrence, the class label will be randomly selected from those labels.

In the above scenario, all the experts are considered equally reliable, as a consequence, even an expert is very confident on its decision, the opinions of less reliable

classifiers may change the final decision. One simple but powerful way of overcoming this drawback is to assign higher weights to the decisions made by more accurate classifiers. So the rule can be rewritten as:

Denote Z the predicted class label, assign $Z \rightarrow \omega_j$ if

$$\Sigma_{i=1}^{N} w_{ik}\delta_{ik} = \max_{j=1}^{C} \Sigma_{i=1}^{N} w_{ij}\delta_{ij}, \tag{4.12}$$

where w_{ij} is the weight of classifier D_i for class ω_j, and δ_{ik} is an indicator function defined as:

$$\delta_{ik} = \begin{cases} 1, & if\ the\ classifier\ D_i\ outputs\ class\ label\ \omega_k \\ 0, & otherwise \end{cases} \tag{4.13}$$

4.6.2 Median Based Fusion

The median based fusion is another popular decision-level fusion method [23]. The idea is to use the median of different classifiers to make the final decision.

Denote Z the predicted class label, $P(\omega_j \mid \mathbf{x}_i)$ the *posteriori* probability of Z assigned as class ω_j by the measurement vector $\mathbf{x}_i$ from the i-th classifier. The median based fusion can be denoted as:

Assign $Z \rightarrow \omega_j$ if

$$\max_{i=1}^{N} P(Z = \omega_j \mid \mathbf{x}_i) = \max_{k=1}^{C} med_{i=1}^{N} P(Z = \omega_k \mid \mathbf{x}_i) \tag{4.14}$$

4.7 Experiments

We conduct experiments on two challenging mice behavior databases, and compare our results with the state-of-the-art methods on each database.

For each action sequence, three features, i.e., STF, DT-Shape, and DT-MBH, are computed, separately. For DT-Shape and DT-MBH features, we employ a bag-of-features scheme. Vocabularies are constructed with the K-means clustering. The RFs method is used for mice action classification.

In the following, we introduce the two databases first, and then present the experimental settings and experimental results. Finally we provide some analysis and discussions of the experimental results.

4.7.1 Databases

In order to evaluate the proposed approach, two mice behavior databases are used in our experiments. The first one is the Caltech Resident-Intruder Mouse dataset (CRIM13) [9], which captures thirteen different social actions between two mice. This database was recorded using two fixed, synchronized cameras at 25fps with a resolution of 640×480, so each scene has both top and side views. Because mice are nocturnal animals, the near-infrared in-cage lighting was used, and the captured videos are monochromatic. The other database is the Mice Behavior Analysis dataset (MBADA) [16, 48], which records the interaction between two/three mice with an infrared camera FLIR A315. The camera has a spatial resolution of 320×240 at 30fps. More details about each database are given below.

CRIM13 database consists of 237×2 videos. Each video lasts ∼10min. The full dataset lasts over 88 hours and has more than 8 millions frames. There are 13 different actions: Approach, Walk away, Circle, Chase, Attack, Copulation, Drink, Eat, Clean, Human, Sniff, Up, and Other. The videos always start with a male "resident" mouse alone in a laboratory enclosure. At some point, a second mouse called "intruder" is introduced into the enclosure by a human and social interaction between the two mice begins. The resident mouse tries to get to know the intruder mouse, thus some behaviors like approach, circle, and sniff will happen at this time. Once it is identified that the intruder is a male mouse, the resident mouse will likely attack it to defend its territory. If the intruder is a female mouse, the resident mouse will likely court her (copulation, chase). Another case is that the resident mouse just ignores the intruder and is engaged in some solitary behaviors like clean, drink, eat, and up. The intruder mouse is removed just before the end of the video. Because of the running time issue, we followed [13] and used a subset of the database: the validation set. This set contains 40 videos (20 from top view and 20 from side view). Half videos are used for training and the other half for testing. We used 10 out of the 13 actions as in [13]. On this database, we follow the same setting as [9, 13] to compute the average per-frame agreement as the error metric, which is computed by taking the average of the diagonal in the confusion matrix. Some example frames from this database are shown in Figure 4.2.

The MBADA dataset is collected for a study of multiple mice tracking and behavior analysis problem. It is composed of 4 subsets: dataA-1, dataA-2, dataA-3, and dataA-4.

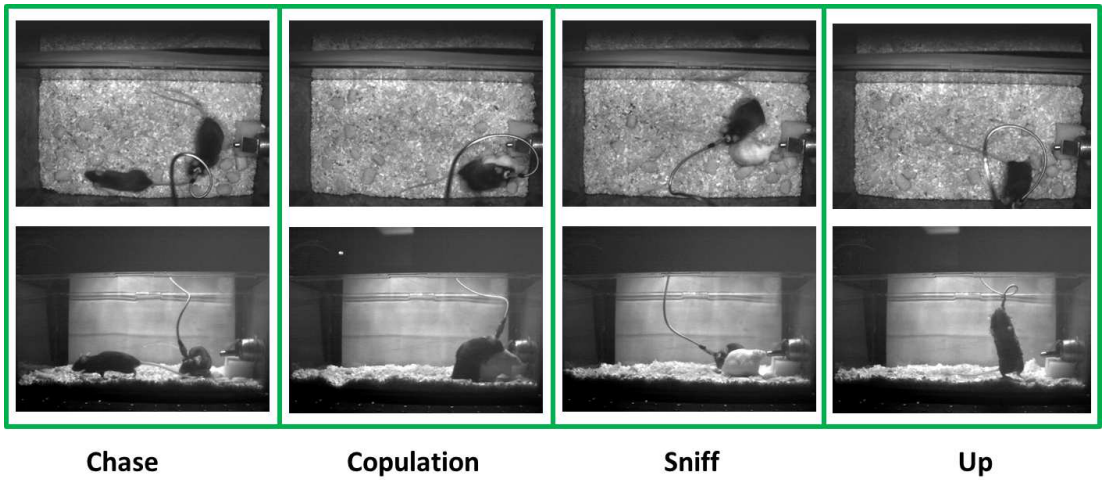

Figure 4.2: Some example frames in the CRIM13 dataset.

The first three sets monitor the social behavior of three interacting mice while dataA-4 monitors two mice. When there are more than two mice, we follow the same setting as [16] and study the behavior between every two mice. Set dataA-1 lasts about 30 minutes, while each of the other three sets lasts about 60 minutes. There are 8 actions in this dataset: Nose2Body, Nose2Nose, Nose2Genitals, Above, Following, StandTogether, StandAlone, and WalkAlone. This dataset was manually labelled by 2 experts: Grader 1 and Grader 2. We checked the two sets of labels and found that Grader 1 has a better labeling. Besides, [16] also shows that when Grader 1 is considered as the ground truth, grader/system's performance is more consistent with the inter-grader one. So in our experiment, we only use the labels from Grader 1. Figure 4.3 shows some example images from this database. Because this database involves behavior analysis among multiple mice, we only focus on two interacting mice and mask out the irrelevant mouse when computing the dense trajectory features.

4.7.2 Experimental Settings

On the CRIM13 database, we follow the same experimental setting as [13]. Specifically, we use the validation set: 10 full videos for training and 10 full videos for testing. On the MBADA database, we follow the same experimental setting as [16]. A 3-fold cross validation approach is adopted. Each video is divided into three consecutive parts

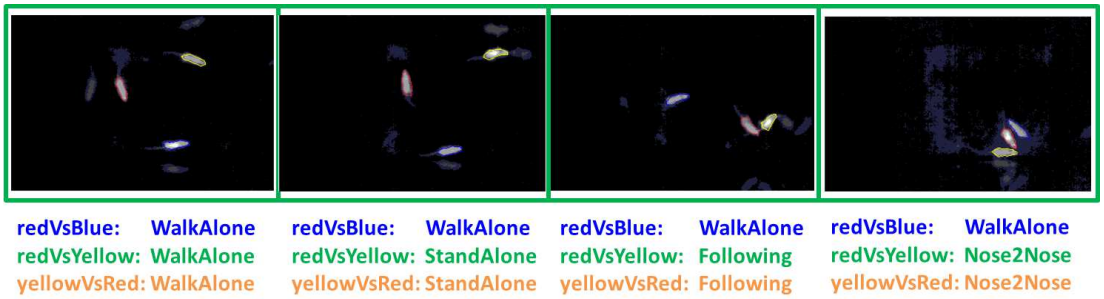

Figure 4.3: Some example images in the MBADA dataset.

(leaving the frame ordering intact). Two folds are joined together for training and the other one for testing. The process is iterated over all folds and the final recognition result is the average of all iterations. The K-means clustering method is applied to quantize the DT-Shape and DT-MBH features into histograms. In our experiment, we set the number of visual words V as 250. To limit the complexity, we cluster a subset of 100,000 randomly selected training features. In order to get the dense trajectory features, we adapted the same setting for human action recognition [57] to the mice actions. The size of space-time volume aligned with trajectory is set as $32 \times 32 \times 15$. To embed structure information, each volume is subdivided into a spatiotemporal grid of size $2 \times 2 \times 3$. For the trajectory length L, we set it to be 9 frames. For temporal coherence features, we used windows of size 75, 185, and 615 to combine short, median, and long term context similar to [9]. For the random forests, the number of trees can be selected from $[1, 500]$, and the number of features used in each split can be selected from $[3, 50]$. To control how deep the tree grows, we set the minimum size of terminal nodes to be 1.

4.7.3 Experimental Results

We present the experimental results of individual features first, so we can understand the performance of each single feature. Then we give the fusion results to measure the improvement.

4.7.3.1 Sparse Trajectory Features

Table 4.1 shows the experimental results using sparse trajectory features(STF) on the two databases and the comparison of recognition performances between sparse trajectory features and the trajectory features(TF) [9]/spatiotemporal features(S-TF) [16]. We can see that STF consistently outperforms both TF and S-TF features. STF achieves an accuracy of 4.80% higher than TF on the CRIM13 database. On the MBADA database, STF outperforms S-TF on each subset. For example, on Set dataA-2_2Vs3, the STF gets the largest improvement of 6.36% over S-TF features. On the whole database, STF improves the average accuracy by 4.31%.

On the CRIM13 database, each mouse is tracked and denoted with 1 point on the body, thus we compute only one distance feature: "body2body", but for the MBADA database, each mouse is denoted with 3 points: head, body, and genital, so we compute five distance features: "body2body", "head2head", "head2body", "head2genital", and "genital2genital". In order to examine whether more distance measures can improve the final recognition result or not, we also compute the sparse trajectory feature with only one distance feature "body2body" for one set from the MBADA database and call this sparse trajectory feature "STF_1Pt". We compare the performances among S-TF, STF, and STF_1Pt on Set dataA-4_2Vs1 in Table 4.2. We can see that STF_1Pt feature can get a 2.87% higher recognition accuracy than S-TF, while the STF can further improve the accuracy over the STF_1Pt by 1.74%.

Table 4.1: Performances of sparse trajectory features (STF) on the two databases and comparison of classification performance between STF and spatiotemporal features(S-TF)[16]/trajectory features(TF)[9]. We show the recognition results after adding temporal coherence feature (TCF).

Dataset		**S-TF / TF + TCF**	**STF + TCF**
CRIM13		42.30%	**47.10%**
MBADA	dataA-1_redVsBlue	76.16%	**81.12%**
	dataA-1_redVsYellow	78.01%	**83.35%**
	dataA-1_yellowVsRed	81.61%	**84.25%**
	dataA-2_2Vs1	69.94%	**72.34%**
	dataA-2_2Vs3	70.94%	**77.30%**
	dataA-3_2Vs1	73.09%	**76.00%**
	dataA-3_2Vs3	73.71%	**79.00%**
	dataA-4_2Vs1	72.06%	**76.67%**
	Average	74.44%	**78.75%**

Table 4.2: Comparison of performances among spatiotemporal features (S-TF), STF, and STF_1Pt on Set dataA-4_2Vs1 of the MBADA database.

Method	Accuracy
S-TF [16]	72.06%
STF_1Pt	74.93%
STF	**76.67%**

4.7.3.2 Dense Trajectory Features

After evaluating STF's performance, we present the results on the two databases using dense trajectory shape (DT-Shape) and dense trajectory motion boundary histograms (DT-MBH) features in Table 4.3. We can see that for both DT-Shape and DT-MBH features, it can improve the recognition accuracy by about 8.00% $\sim$ 9.00% using the temporal coherence features. We also observe that DT-MBH consistently outperforms the DT-Shape descriptor. On the CRIM13 database, top view can get slightly higher accuracy than side view.

Table 4.3: Performances of dense trajectory shape (DT-Shape) and motion boundary histograms (DT-MBH) features on the CRIM13 and MBADA databases. Top means top view, and Side means side view.

Dataset		Accuracy			
		Without TCF		With TCF	
		DT-Shape	DT-MBH	DT-Shape	DT-MBH
CRIM13	Top	28.50%	31.70%	39.10%	40.50%
	Side	25.80%	27.10%	33.60%	37.40%
MBADA	dataA-1_redVsBlue	51.34%	58.86%	59.66%	65.94%
	dataA-1_redVsYellow	50.09%	66.76%	60.45%	72.79%
	dataA-1_yellowVsRed	55.03%	68.11%	64.58%	72.92%
	dataA-2_2Vs1	46.29%	53.24%	56.02%	62.14%
	dataA-2_2Vs3	44.29%	52.34%	56.54%	60.12%
	dataA-3_2Vs1	47.57%	61.00%	58.68%	67.55%
	dataA-3_2Vs3	51.67%	57.99%	60.59%	64.63%
	dataA-4_2Vs1	41.35%	50.29%	50.31%	59.79%
	Average	48.45%	58.57%	58.35%	65.74%

4.7.3.3 Comparison of Dense Trajectory Features with STIP features

From Table 4.1 and Table 4.3, we can see that dense trajectory features get lower accuracy than sparse trajectory features, but can dense trajectory features get higher accuracy than another popular category of spatiotemporal descriptor: spatiotemporal interest points (STIP) [12, 24, 28, 30]? To answer this question, we compare dense trajectory features with some popular STIP features on the CRIM13 database in Table 4.4. We can see that among different detector and descriptor combinations, Cuboids+Cuboids can get the highest accuracy of 24.6% on this database. However, both DT-Shape and DT-MBH features can get higher accuracies than the Cuboids+Cuboids feature. This comparison tells us that the dense trajectory features perform better than the popular STIP features in mice behavior understanding.

Table 4.4: Comparison of dense trajectory features (without temporal coherence features) with STIPs features (without temporal coherence features) on the CRIM13 database.

Method		**Top**	**Side**
Dense Trajectory Features	DT-Shape	**28.5%**	25.8%
	DT-MBH	**31.7%**	27.1%
STIPs	Harris3D+Cuboids	20.9%	-
	Harris3D+HOG3D	18.7%	-
	Harris3D+HOG/HOF	15.5%	-
	Cuboids+Cuboids	24.6%	-
	Cuboids+HOG3D	18.2%	-
	Cuboids+HOG/HOF	19.8%	-

4.7.3.4 Fusing Sparse Trajectory Features with Dense Trajectory Features

In this task, we study whether the dense trajectory features can be used to further boost the performance of sparse trajectory features. We combine the sparse trajectory features and dense trajectory features with fusion schemes, e.g., the majority voting and median based fusion.

The experimental results on the CRIM13 database are shown in Table 4.5. Based on fusion, the recognition rate can be improved to 52.40% by majority voting and 56.20% by median based fusion. Both are higher than any individual feature, e.g., 39.10% by DT-Shape, 40.50% by DT-MBH, and 47.10% by STF, respectively. This result validates our proposed approach, i.e., dense trajectory features can be supplementary to sparse

Table 4.5: The recognition accuracies of fusing sparse and dense trajectory features on the CRIM13 database. MAJ and MED denote majority voting and median based fusion, respectively. Note that we only use dense trajectory features from top view because top view gets higher accuracies than side view.

Method		**Accuracy**
Single Feature	DT-Shape Top	39.10%
	DT-MBH Top	40.50%
	STF	47.10%
Fusion	MAJ	**52.40%**
	MED	**56.20%**

trajectory features, thus it can improve the recognition accuracy. The confusion matrix by median based fusion is shown in Figure 4.4.

Table 4.6: The recognition accuracies of fusing sparse and dense trajectory features on the MBADA database.

Set	**Accuracy**				
	Single Feature			**Fusion**	
	DT-Shape	**DT-MBH**	**STF**	**MAJ**	**MED**
dataA-1_redVsBlue	59.66%	65.94%	81.12%	**83.15%**	81.61%
dataA-1_redVsYellow	60.45%	72.79%	83.35%	86.07%	**86.97%**
dataA-1_yellowVsRed	64.58%	72.92%	84.25%	**87.07%**	86.19%
dataA-2_2Vs1	56.02%	62.14%	72.34%	**77.47%**	74.02%
dataA-2_2Vs3	56.54%	60.12%	77.30%	**82.22%**	79.70%
dataA-3_2Vs1	58.68%	67.55%	76.00%	**80.98%**	79.20%
dataA-3_2Vs3	60.59%	64.63%	79.00%	**82.10%**	81.15%
dataA-4_2Vs1	50.31%	59.79%	76.67%	**79.78%**	79.50%
Average	58.35%	65.74%	78.75%	**82.36**%	81.04%

The fusion results on the MBADA dataset are shown in Table 4.6. We can see that after fusing the two features, the average accuracy achieves to 82.36%, which is significantly higher than any single feature. For example, the accuracy of the best individual feature STF is 78.75%. Between the two fusion methods,i.e.,MAJ, and MED, MAJ performs better than MED on almost all subsets except dataA1_redVsYellow. On Set dataA1_redVsYellow, MED can get an accuracy of 86.97%, a little higher than 86.07% by MAJ. Figure 4.5 shows confusion matrix by MAJ on Set dataA1_yellowVsRed, where most of the actions are separated reasonably well.

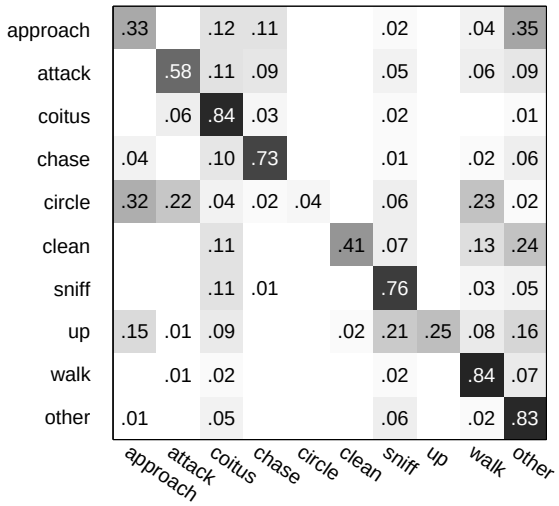

Figure 4.4: The confusion matrix of median based fusion on the CRIM13 Database.

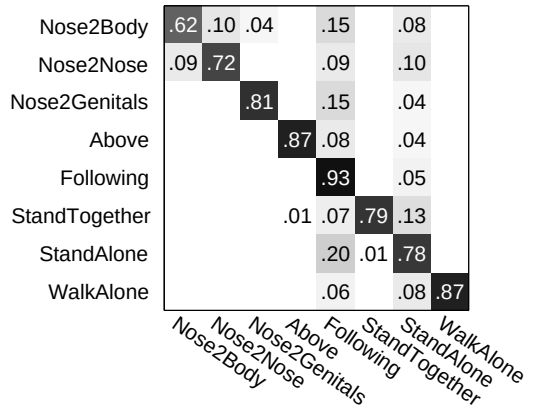

Figure 4.5: The confusion matrix of majority voting on the Set dataA1_yellowVsRed ("yellow" means the mouse labeled as "yellow", "red" means the mouse labeled as "red").

Table 4.7: Comparison of the recognition accuracies between our approaches and all the state-of-the-art methods on the CRIM13 and MBADA databases.

Method	Accuracy	
	CRIM13	**MBADA**
TF + Structured SVM [13]	37.20%	-
TF [9]	42.30%	72.26%
S-TF [16]	-	74.44%
STF (Ours)	**47.10%**	**78.75%**
Combined Features by MAJ (Ours)	**52.40%**	**82.36%**
Combined Features by MED (Ours)	**56.20%**	**81.04%**

4.7.3.5 Computation Complexity

We report run time (frames per second) of both sparse and dense trajectory features, and also compare with some popular STIP features in Table 4.8. The run-time is obtained on a Dell desktop with a 3.4 GHz Intel Core i7-2600 CPU and 12GB RAM. For the STF feature, we implement it in Matlab (given the tracks, available from project website [8]). For DTF, we use the toolbox from [58] under our settings, for example, the trajectory length is set to be 9. As a comparison, we also compute STIP (Harris3D+HOGHOF and Cuboids+Cuboids) features with the toolboxes provided by [30] and [12], respectively.

From the results, we can see that the sparse trajectory feature achieves real-time with a speed of greater than 1.0×10^3fps. Note that the STF feature only relies on mice locations:(x, y), then some geometry features like distance, velocity are computed. The database website provides mice locations before hand, so I do not need to read/process image for the computation of STF, I only need to load in mice locations in matlab and then do some calculation on those (x, y) locations. Also the 1.0×10^3fps is only for STF feature extraction. The time for classification is not counted, because feature extraction is the most time-consuming part. Cuboids + Cuboids feature's run time is 10.0fps, Harris3D + HOG/HOF can get 2.3fps. Dense trajectory feature gets a speed of 1.1fps, which is slower than STIP features. But in Section 4.7.3.3, we have shown that dense trajectory feature can get better performance than all STIP features.

Table 4.8: Comparison of different features' computation complexity on the CRIM13 dataset. fps denotes frames/second. *Mice locations are given before hand, no image/video processing was performed. The other features involve image/video processing.

Feature		**fps**
STF		$> 1.0 \times 10^{3*}$
DTF		1.1
STIP	Cuboids+Cuboids	10.0
	Harris3D+HOG/HOF	2.3

4.7.4 Comparison with the State-of-the-Art Methods

We further compare our approach with the state-of-the-art methods for mice behavior recognition on the two challenging datasets in Table 4.7. We list all the published results on the two databases, to the best of our knowledge. On the CRIM13 database, the current state-of-the-art performance on this dataset is 42.30%, while our approach can improve the accuracy to 52.40%, and 56.20% by median based fusion. On the MBADA dataset, STF can get an accuracy of 78.88%, which is higher than both [9] and [16]. After fusion with MAJ and MED, the accuracy can be further improved to 82.36% and 81.04%, respectively as shown in Table 4.7. Therefore, our approach can have significantly better results than state-of-the-art methods.

4.8 Conclusions

We have presented a new approach to mice behavior recognition, based on integration of sparse and dense trajectory features using different fusion methods. Comprehensive experiments have been conducted on two mice behavior databases. We have shown that the sparse trajectory features can perform better than dense trajectory features. Fusion of these two kinds of features can improve the performance significantly. We have also shown that the proposed approach outperforms the state-of-the-art methods on both datasets.

5

Conclusion & Future Work

We have presented our approaches for human action recognition using RGB-D data, and mice behavior recognition based on the integration of sparse and dense trajectory features.

We have proposed an effective framework for depth-based human action recognition. Five different features are investigated under our framework. Two of the five features, i.e., skeleton trajectory shape descriptor (ST-Shape) and skeleton trajectory motion boundary histogram features (ST-MBH) are proposed by us for posture and motion representation in depth data. The top three best features are combined by a probabilistic fusion approach (PFA). The experimental results demonstrate that the TriViews framework is very effective to improve the RGB-D human action recognition performance, outperforming the state-of-the-art results on each of the three challenging databases.

We have also presented a new approach for mice behavior recognition, based on the fusion of sparse and dense trajectory features. Experimental results show that sparse trajectory features can perform better than dense trajectory features, however, fusion of the two features can improve the performance significantly. Our approach is validated on two public mice databases and gets the state-of-the-art performance on both databases.

In the future, we will consider the following work:

(1) Because RGB-D data have many advantages over traditional gray-level images/videos, we will consider collecting an animal behavior database with the Kinect. And do some research on that database. (2) In our TriViews framework, we currently

treat all three views equally. However, different views contain different amount of information, so it is interesting to develop some novel approaches to handle different views in different ways. For example, one very simple way is to assign different weights to different views when combining the three views. (3) Design and develop some new features for human/animal behavior representation, and investigate the new features under our TriViews framework for human action recognition, and fuse the new features for our mice behavior recognition work.

Bibliography

[1] J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43(3):16:1–16:43, 2011.

[2] F. Alkoot and J. Kittler. Experimental evaluation of expert fusion strategies. *Pattern Recognition Letters*, 20(11):1361–1369, 1999.

[3] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. In *ICML*, volume 3, pages 3–10, 2003.

[4] S. Azary and A. Savakis. Grassmannian sparse representations and motion depth surfaces for 3d action recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, pages 492–499, 2013.

[5] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conf. on Computer Vision*, pages 404–417. Springer, 2006.

[6] S. Belongie, K. Branson, P. Dollár, and V. Rabaud. Monitoring animal behavior in the smart vivarium. *Workshop on Measuring Behavior*, pages 70–72, 2005.

[7] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[8] X. Burgos-Artizzu. Caltech Resident-Intruder Mouse dataset. `http://www.vision.caltech.edu/Video_Datasets/CRIM13/CRIM13/Main.html`, 2012.

[9] X. Burgos-Artizzu, P. Dollár, D. Lin, D. Anderson, and P. Perona. Social behavior recognition in continuous video. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pages 1322–1329, 2012.

[10] A. Chaaraoui, J. Padilla-López, P. Climent-Pérez, and F. Flórez-Revuelta. Evolutionary joint selection to improve human action recognition with rgb-d devices. *Expert Systems with Applications*, 41(3):786–794, 2014.

[11] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, A. Del Bimbo, et al. Space-time pose representation for 3d human action recognition. In *ICIAP Workshop on Social Behaviour Analysis*, pages 456–464. Springer Berlin Heidelberg, 2013.

[12] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *2nd Joint IEEE Int'l Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

[13] E. Eyjolfsdottir, X. Burgos-Artizzu, S. Branson, K. Branson, D. Anderson, P. Perona, and H. Farm. Learning animal social behavior from trajectory features. 2012.

[14] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis*, pages 363–370. Springer, 2003.

[15] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[16] L. Giancardo, D. Sona, H. Huang, S. Sannino, F. Managò, D. Scheggia, F. Papaleo, and V. Murino. Automatic visual tracking and social behaviour analysis with multiple mice. *PLOS ONE*, 8(9), 2013.

[17] G.-D. Guo, Y. Fu, C. Dyer, and T. Huang. A probabilistic fusion approach to human age prediction. In *Computer Vision and Pattern Recognition Workshops, IEEE Computer Society Conference on*, pages 1–6, 2008.

[18] D. O. Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[19] H. Jhuang, E. Garrote, X. Yu, V. Khilnani, T. Poggio, A. Steele, and T. Serre. Automated home-cage behavioural phenotyping of mice. *Nature communications*, 1:68, 2010.

[20] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *IEEE Int'l Conf. on Computer Vision*, pages 1–8, 2007.

[21] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. In *Computer Vision–ECCV*, pages 425–438. Springer, 2012.

[22] S. Joshi, E. Klassen, A. Srivastava, and I. Jermyn. A novel representation for riemannian analysis of elastic curves in rn. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–7, 2007.

[23] J. Kittler. Combining classifiers: A theoretical framework. *Pattern analysis and Applications*, 1(1):18–27, 1998.

[24] A. Klaser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *British Machine Vision Conference*, pages 99.1–99.10. BMVA Press, 2008.

[25] H. Koppula, R. Gupta, and A. Saxena. Learning human activities and object affordances from rgb-d videos. 2012.

[26] L. Kuncheva. A theoretical study on six classifier fusion strategies. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2):281–286, 2002.

[27] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[28] I. Laptev and T. Lindeberg. Space-time interest points. In *Ninth IEEE Int'l Conf. on Computer Vision*, pages 432–439, 2003.

[29] I. Laptev and T. Lindeberg. Velocity adaptation of space-time interest points. In *Proc. of the 17th Int'l Conf. on Pattern Recognition*, volume 1, pages 52–56, 2004.

[30] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[31] M. Leandro, V. Thales, M. Dimas, L. Thomas, W. Antonio, and F. C. Mario. Real-time gesture recognition from depth data through key poses learning and decision forests. *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, 0:268–275, 2012.

[32] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 9–14, 2010.

[33] D. Lin, M. Boyle, P. Dollar, H. Lee, E. Lein, P. Perona, and D. Anderson. Functional identification of an aggression locus in the mouse hypothalamus. *Nature*, 470(7333):221–226, 2011.

[34] L. Liu and L. Shao. Learning discriminative representations from rgb-d video data. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1493–1500. AAAI Press, 2013.

[35] B. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, 1981.

[36] R. Messing, C. Pal, and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *Computer Vision, 12th International Conference on*, pages 104–111. IEEE, 2009.

[37] C. Mombereau, K. Kaupmann, W. Froestl, G. Sansig, H. van der Putten, and J. Cryan. Genetic and pharmacological evidence of a role for gaba b antidepressant-like behavior. *Neuropsychopharmacology*, 29(6), 2004.

[38] B. Ni, G. Wang, and P. Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. In *Computer Vision Workshops (ICCV Workshops)*, pages 1147–1153. IEEE, 2011.

[39] E. Ohn-Bar and M. Trivedi. Joint angles similiarities and hog2 for action recognition, 2013.

[40] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. pages 716–723, 2013.

[41] E. Pekalska. *Dissimilarity representations in pattern recognition. Concepts, theory and applications*. Ph. D. thesis, Delft University of Technology, Delft, The Netherlands, 2005.

[42] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[43] M. Raptis and S. Soatto. Tracklet descriptors for action modeling and video analysis. In *Computer Vision–ECCV*, pages 577–590. Springer, 2010.

[44] L. Seidenari, V. Varano, S. Berretti, P. Pala, and A. Del Bimbo. Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses, 2013.

[45] A. Shabani, D. Clausi, and J. Zelek. Evaluation of local spatio-temporal salient feature detectors for human action recognition. In *9th Conf. on Computer and Robot Vision (CRV)*, pages 468–475, 2012.

[46] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. pages 1–8, 2011.

[47] H. Soh and Y. Demiris. Iterative temporal learning and prediction with the sparse online echo state gaussian process. In *Neural Networks (IJCNN), International Joint Conference on*, pages 1–8. IEEE, 2012.

[48] M. Soto. Mice Behaviour Analysis - Datasets. `http://www.iit.it/en/datasets-and-code/1662-micedataset.html`, 2013.

[49] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal

context modeling for action recognition. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 2004–2011, 2009.

[50] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from rgbd images. In *IEEE Int'l Conf. on Robotics and Automation (ICRA)*, pages 842–849, 2012.

[51] L. Tecott and E. Nestler. Neurobehavioral assessment in the information age. *Nature neuroscience*, 7(5):462–466, 2004.

[52] I. Theodorakopoulos, D. Kastaniotis, G. Economou, and S. Fotopoulos. Pose-based human action recognition via sparse representation in dissimilarity space. *Journal of Visual Communication and Image Representation*, 25(1):12–23, 2013.

[53] P. Turaga, R. Chellappa, V. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, 2008.

[54] A. Vieira, E. Nascimento, G. Oliveira, Z. Liu, and M. Campos. Stop: Space-time occupancy patterns for 3d action recognition from depth map sequences. *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 252–259, 2012.

[55] E. Vig, M. Dorr, and D. Cox. Space-variant descriptor sampling for action recognition based on saliency and eye movements. In *Computer Vision–ECCV*, pages 84–97. Springer, 2012.

[56] C. Wang, Y. Wang, and A. Yuille. An approach to pose-based action recognition. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 915–922, 2013.

[57] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pages 3169–3176, 2011.

[58] H. Wang, C. Schmid, et al. Action recognition with improved trajectories. In *International Conference on Computer Vision*, 2013.

[59] H. Wang, M. Ullah, A. Kläser, I. Laptev, C. Schmid, et al. Evaluation of local spatio-temporal features for action recognition. In *BMVC 2009-British Machine Vision Conference*, pages 124.1–124.11, 2009.

[60] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. In *Computer Vision–ECCV*, pages 872–885. 2012.

[61] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1290–1297, 2012.

[62] D. Weinland, R. Ronfard, and E. Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2):224–241, 2011.

[63] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *European Conf. on Computer Vision*, pages 650–663, 2008.

[64] L. Xia and J. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. *IEEE Conf. on Computer Vision and Pattern Recognition*,

pages 2834–2841, 2013.

[65] L. Xia, C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 20–27, 2012.

[66] X. Yang and Y. Tian. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 14–19, 2012.

[67] X. Yang and Y. Tian. Effective 3d action recognition using eigenjoints. *Journal of Visual Communication and Image Representation*, 2013.

[68] X. Yang, C. Zhang, and Y. Tian. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 1057–1060. ACM, 2012.

[69] H. Zhang and L. Parker. 4-dimensional local spatio-temporal features for human activity recognition. In *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, pages 2044–2049, 2011.

[70] Y. Zhao, Z. Liu, L. Yang, and H. Cheng. Combing rgb and depth map features for human activity recognition. In *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), Asia-Pacific*, pages 1–4, 2012.

[71] Y. Zhu, W. Chen, and G.-D. Guo. Fusing spatiotemporal features and joints for 3d action recognition. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 486–491, 2013.

[72] Y. Zhu, W. Chen, and G.-D. Guo. Evaluating spatiotemporal interest point features for depth-based action recognition. *Image and Vision Computing*, 2014.